François de Dieuleveult / Hervé Fanet

# Principes et pratique de l'électronique

## Tome 2

## Fonctions numériques & mixtes

DUNOD

François de Dieuleveult Hervé Fanet

# Principes et pratique de l'électronique

## Tome 2

## Fonctions numériques et mixtes

DUNOD

Ce pictogramme mérite une explication. Son objet est d'alerter le lecteur sur la menace que représente pour l'avenir de l'écrit, particulièrement dans le domaine de l'édition technique et universitaire, le développement massif du **photocopillage**.

Le Code de la propriété intellectuelle du 1er juillet 1992 interdit en effet expressément la photocopie à usage collectif sans autorisation des ayants droit. Or, cette pratique s'est généralisée dans les établissements d'enseignement supérieur, provoquant une baisse brutale des achats de livres et de revues, au point que la possibilité même pour les auteurs de créer des œuvres nouvelles et de les faire éditer correctement est aujourd'hui menacée.

Nous rappelons donc que toute reproduction, partielle ou totale, de la présente publication est interdite sans autorisation du Centre français d'exploitation du droit de copie (**CFC**, 20 rue des Grands-Augustins, 75006 Paris).

© Dunod, Paris, 1997
ISBN 2 10 003321 2

Toute représentation ou reproduction intégrale ou partielle faite sans le consentement de l'auteur ou de ses ayants droit ou ayants cause est illicite selon le Code de la propriété intellectuelle (Art L 122-4) et constitue une contrefaçon réprimée par le Code pénal. • Seules sont autorisées (Art L 122-5) les copies ou reproductions strictement réservées à l'usage privé du copiste et non destinées à une utilisation collective, ainsi que les analyses et courtes citations justifiées par le caractère critique, pédagogique ou d'information de l'œuvre à laquelle elles sont incorporées, sous réserve, toutefois, du respect des dispositions des articles L 122-10 à L 122-12 du même Code, relative à la reproduction par reprographie.

# TOME 2 : FONCTIONS NUMÉRIQUES ET MIXTES

## PARTIE 1 : LES CIRCUITS LOGIQUES


**■ Chapitre I : Logique combinatoire**

***États logiques*** 4
***Portes logiques*** 7
***Algèbre de Boole*** 12
1) Règles générales 12
2) Règles particulières 12
3) Simplification des expressions logiques 13
***Représentations binaire, hexadécimale et ASCII*** 16
1) Représentation binaire des nombres 16
2) Représentation hexadécimale 18
3) Le code ASCII 18
***Arithmétique binaire*** 19
***Exemples de fonctions combinatoires*** 21
1) Démultiplexeur et multiplexeur logique 21
2) Commande d'afficheur 24
3) Circuits programmables 27
4) Les mémoires ROM 29
***Entraînement*** 30

**■ Chapitre II : Logique séquentielle**

***Bascule RS*** 31
1) Bascule RS avec portes NAND 32
2) Bascule RS avec portes NOR 32
3) Bascule RS synchronisée 34
***Bascule D*** 35
1) Bascule maître-esclave déclenchée par les fronts 35
2) Entrées de prépositionnement et remise à zéro 37
***Bascule JK maître-esclave*** 38
1) Limitation en fréquence et temps de propagation 41
***Registres à décalage*** 42
1) Registre à entrée série et sortie parallèle 43
2) Registre à entrée série et sortie série 44
3) Registre à entrées parallèles et sortie série 45
4) Registre à entrées parallèles et sorties parallèles 45
5) Registre à décalage universel 45
***Applications des registres à décalage*** 47
1) Circuit de retard numérique 47
2) Générateur pseudo-aléatoire 47
***Les compteurs-décompteurs*** 55
1) Compteurs synchrones 55
2) Compteurs asynchrones 56
3) Synthèse des compteurs synchrones 56
4) Synthèse des compteurs asynchrones 59

5) Compteurs modulo *N* ou diviseur par *N* 61
6) Exemples de décompteurs 62
***Applications des compteurs*** 64
1) Comptage direct 64
2) Diviseur par *N* 65
3) Mesure d'une fréquence 66
4) Mesure de temps 67
5) Mesure d'une distance 68
6) Mesure de vitesse 68
***Description des systèmes de logique séquentielle en machines d'états*** 69
1) Exemple de la bascule JK 69
2) Exemple des feux de circulation 70
3) Synthèse des fonctions logiques 72
***Entraînement*** 76

**■ Chapitre III : Les familles logiques**

***Les propriétés des circuits logiques*** 79
1) Marges de bruit et précautions à prendre 79
2) Temps de commutation et temps de propagation 82
3) Temps de *hold* et de *set-up* 83
4) Entrance et sortance : *Fan-in* et *Fan-out* 85
5) Fonctions et réalisation d'un *ET* câblé 86
6) Consommation statique et dynamique 87
7) Logique à trois états 88
***Les familles logiques*** 90
1) Logique TTL 90
2) Logique ECL 94
3) Logique CMOS et BICMOS 97
***Les mémoires*** 100
1) Mémoire et adressage 101
2) *Random Access Memory* (RAM) 105
3) *Read Only Memory* (ROM), EPROM et EEPROM 106
4) Mémoire flash 108
5) Mémoires magnétiques et optiques 109
***Entraînement*** 112

**■ Chapitre IV : Microcontrôleurs, microprocesseurs et micro-ordinateurs**

***Définition des termes*** 113
1) Largeur du mot 113
2) CPU (*Central Processing Unit*) 114
3) Les mémoires 114
4) Le microcontrôleur 114
5) Le microprocesseur 115
6) Le micro-ordinateur 116
***Exemple d'un microcontrôleur : description d'un 8051*** 117
1) Généralités sur le microcontrôleur de base 118
2) Les mémoires du microcontrôleur 121
3) Caractéristiques du microcontrôleur 8 bits 8051 122
4) Configuration hardware du microcontrôleur 125
5) La mémoire RAM en détail 134
6) Les registres spéciaux SFR 134
7) Circuit d'horloge et oscillateur 136
8) Timing du CPU 136

9) Les opérations d'entrées/sorties 137
10) Écriture d'un port 138
11) Compteur/temporisateur (*Timer/event Counter*) 138
12) Interface série 140
13) Vitesses de transfert ou Baud Rates 141
14) Les interruptions 143
15) Application d'un 8751 à la programmation de deux PLL 146
***Exemple d'un microprocesseur*** 155
1) Architecture système 157
2) Structure interne du processeur 80286 158
3) Segmentation de la mémoire 160
4) Les registres du 80286 162
5) Quelques instructions assembleur de la famille 8086 165
***Le micro-ordinateur : architecture, caractéristiques et utilisation*** 166
1) CPU ou processeur 166
2) Mémoires 166
3) Mémoire de masse 167
4) Clavier, écran et souris 167
5) Imprimantes et traceurs 168
6) Interface d'entrée/sortie 170
7) Interface réseau 171

**■ Chapitre V : Console de sélection vidéo à microcontrôleur**

***Sélection de la fonction à accomplir*** 173
***Choix et description des composants*** 174
1) Schéma synoptique du commutateur 174
2) Le circuit commutateur SONY CXA 1114 176
3) Le circuit SAA 5244 A 178
4) Choix du microcontrôleur et de la structure 179
***Schéma de principe*** 180
1) Microcontrôleur 180
2) Incrustation 181
3) Commutateur CXA 1114 181
4) Télécommande IR 181
***Manuel d'utilisation*** 184

**PARTIE 2 : LES CIRCUITS ÉLECTRONIQUES COMPLEXES**

**■ Chapitre VI : Sources stabilisées et alimentation des systèmes**

***Utilisation des sources de tension stabilisées*** 189
***Principe de fonctionnement des sources de tension stabilisées*** 191
***Exemple simple d'alimentation*** 193
***Alimentation à découpage et convertisseur continu-continu*** 195
***Caractéristiques des régulateurs et critères de choix*** 199
1) Régulateurs linéaires 199
2) Régulateurs à découpage 199

**■ Chapitre VII : Les conversions analogique-numérique et numérique-analogique**

***Conversion analogique-numérique et acquisition de données*** 202
***Choix de la fréquence d'échantillonnage*** 205
***Choix du nombre de bits de codage*** 209
***Convertisseur parallèle*** 213
***Convertisseur à approximations successives*** 216

***Convertisseur à rampe*** 220
***Convertisseur sigma-delta*** 222
***Récapitulatif des convertisseurs analogique-numérique*** 222
***Convertisseur numérique-analogique*** 224
***Interfaces avec les microprocesseurs*** 227
1) Étape 1 : Échantillonnage du signal 230
2) Étape 2 : Démarrage de la conversion 231
3) Étape 3 : Lecture de la valeur de l'échantillon par le processeur 231
***Caractéristiques des convertisseurs et critères de choix*** 232
1) Précision (*accuracy*) 232
2) Nombre de conversions par seconde ou fréquence d'échantillonnage (*sampling frequency*) 232
3) Rapport signal sur bruit et bits effectifs (*signal to noise ratio and effective bits*) 232
4) Gamme du signal d'entrée (*range*) 232
5) Précision d'échantillonnage (*aperture jitter*) 232
6) Non linéarité intégrale et non linéarité différentielle (*integral non linearity, differential non linearity*) 233
7) Consommation (*power consumption*) 233
***Quelques exemples de produits*** 234
***Entraînement*** 235

**■ Chapitre VIII : Synthétiseur de fréquence**

***Analyse qualitative du fonctionnement de la boucle*** 238
1) Principe 239
***Le filtre de boucle*** 244
1) Saut de phase 245
2) Saut de fréquence 246
***Plage de capture et plage de verrouillage*** 250
***Modulation d'une boucle à verrouillage de phase*** 251
1) Modulation de phase 251
2) Modulation de fréquence 252
***Démodulation par une boucle à verrouillage de phase*** 253
***Bruit de phase*** 254
***Éléments constituant une boucle à verrouillage de phase*** 255
1) VCO 256
2) Détecteur de phase 260
3) Oscillateurs à quartz de référence 266

**■ Chapitre IX : Modems et transmissions numériques**

***Milieu de transmission*** 270
***Problème lié au signal NRZ*** 271
***Transmission en bande de base*** 273
1) Probabilité ou taux d'erreur bit 273
2) Code biphase 275
3) Code bipolaire 277
4) Code à haute densité HDB3 278
***Modulation autour d'une fréquence porteuse*** 279
1) Modulation d'amplitude par un signal tout ou rien 279
2) Modulation par déplacement de fréquence FSK 281

**■ Chapitre X : Circuits d'interface pour bus et réseaux**

***Liaison série et liaison parallèle, bus et réseaux*** 286
1) Liaison série et liaison parallèle 286
2) Sérialisation et désérialisation des données par un registre à décalage 290
3) Bus et réseaux 291

***Exemple du bus PC*** 292
1) Le bus PC 292
2) Description de l'interface d'une carte PC oscilloscope numérique 296
3) Le traitement des interruptions 297
4) Tableau résumé des lignes du bus PC/AT 301
***Transmission synchrone ou asynchrone*** 302
***Les standards de communication*** 305
1) Les bus fond de panier 307
2) Liaisons série et parallèle avec périphérique éloigné 308
3) Les réseaux locaux 312
***Les UART (Universal Asynchronous Receivers Transmitters)*** 313

**■ Chapitre XI : Filtres à capacités commutées et filtres numériques**

***Limitations dues aux structures classiques*** 318
***Intégrateur à capacité commutée*** 319
***Filtre à variable d'état*** 320
1) Calcul de la fonction de transfert du filtre à variable d'état 320
2) Filtre réjecteur 323
3) Limites du système 323
***Entraînement*** 325

**■ Chapitre XII : Carte d'acquisition vidéo pour PC**

***Définition du signal vidéocomposite*** 329
***La numérisation vidéo*** 329
1) Décodeur PAL/SECAM 332
2) Schéma de principe de la carte de numérisation et acquisition 332
3) Utilisation des adresses 346
4) Mémorisation de l'image 348
5) Conversion D-A 349
6) Filtrage des signaux de sortie 354
7) Programmation de la carte 356
8) Programme d'écriture en mémoire 359

**Conclusion** 361

**Index** 373

**Bibliographie** 375

# 1 LES CIRCUITS LOGIQUES


1. Logique combinatoire
2. Logique séquentielle
3. Les familles logiques
4. Microprocesseurs, microcontrôleurs et micro-ordinateurs
5. Application : la console de sélection vidéo à microcontrôleur


Les circuits logiques occupent une place de tout premier plan dans l'électronique actuelle. Ils permettent d'effectuer des opérations mathématiques ou logiques sur les représentations binaires de grandeurs. Ces grandeurs sont des grandeurs physiques dans le contrôle des procédés mais ce sont également les valeurs obtenues par le codage des sons et des images dans les systèmes de transmission. Pensons au téléphone, à l'enregistrement des disques laser et à la télévision numérique de demain pour mesurer l'importance des techniques numériques en électronique.
Les deux premiers chapitres de cette partie indiquent les principes généraux des systèmes logiques : combinatoires quand le temps n'intervient pas, et séquentiels quand les états se modifient dans le temps.
Le chapitre sur les familles logiques montre comment ces fonctions sont réalisées pratiquement et donne des critères sur le choix des technologies.
Enfin un important chapitre est consacré à la description des composants les plus complexes de l'électronique : les microprocesseurs. Ces circuits permettent de dérouler un programme préalablement écrit. Leur importance est considérable et ils ont permis de mettre sur le marché un appareil de large diffusion, le micro-ordinateur.

# LOGIQUE COMBINATOIRE

## SOMMAIRE


- États logiques
- Portes logiques
- Algèbre de Boole
- Représentations binaire, hexadécimale et ASCII
- Arithmétique binaire
- Exemples de fonctions combinatoires
- Entraînement


*L'électronique dite **logique** constitue actuellement la partie la plus importante des systèmes électroniques. Sa particularité tient au fait que les circuits logiques ne savent reconnaître que deux états possibles. Ces deux états seront appelés **niveau bas** et **niveau haut** ou **niveau zéro** et **niveau un**. Cette particularité semble limitative mais il suffit de penser que tout nombre peut, par un codage en base 2, se représenter par une suite formée à partir de deux symboles, pour imaginer les possibilités des circuits logiques comme outils de calcul. Ce point sera repris dans le paragraphe consacré à l'**arithmétique binaire**. De plus, tout symbole, chiffre ou lettre, peut se coder par une suite de 0 et de 1, comme il sera vu dans le paragraphe consacré aux représentations. Le traitement de l'information, c'est-à-dire la création de messages, leur transport et l'extraction à l'arrivée de la partie signifiante, peut alors s'effectuer en utilisant des circuits logiques. Le secteur des télécommunications et des liaisons numériques entre calculateurs est donc directement concerné par ces techniques.*

*Le circuit élémentaire est la **porte logique**. Elle permet, en fonction des états des entrées, de fournir une sortie dans un des deux états logiques possibles. Ce chapitre montre qu'à partir de quelques types de portes, il est possible de réaliser les fonctions logiques les plus complexes. Les circuits intégrés actuels peuvent comporter plusieurs millions de portes. C'est par exemple le cas des microprocesseurs qui sont capables de mettre en œuvre l'arithmétique binaire à partir d'un **programme**.*

*Dans un premier temps, nous étudierons la **logique combinatoire**. Elle permet, dans un système électronique, de définir les états d'un certain nombre de sorties en fonction des états des entrées et d'eux seuls. Dans un deuxième temps, ce type de système sera généralisé pour donner lieu à la **logique séquentielle**. Dans ce cas, l'état d'un système dépend également de la séquence logique des états précédents, entrées et sorties.*

*Enfin, il faut garder en mémoire que l'électronique logique est un cas particulier de l'électronique analogique dans la mesure où les signaux restent des signaux électriques créés par des composants actifs et passifs. Les propriétés électriques des circuits logiques feront donc référence aux propriétés étudiées dans les chapitres précédents.*

## États logiques

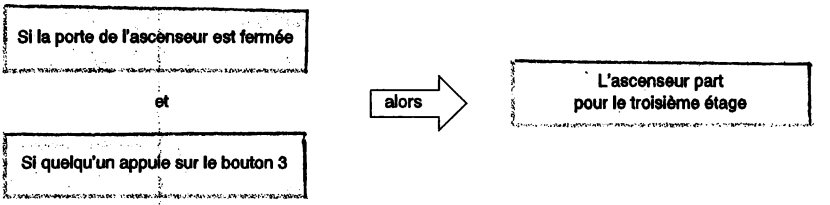
Dans les sciences mathématiques, la logique désigne l'ensemble des relations entre des propositions supposées vraies ou fausses. Par exemple, on écrira pour la fonction « ET » : si (*p*) est vraie **ET** si (*q*) est vraie, alors la proposition (*r*) est vraie. Dans la vie quotidienne, des raisonnements du type suivant sont tenus : si la porte de l'ascenseur est fermée et si quelqu'un appuie sur le bouton 3, alors l'ascenseur part au troisième étage. Il suffit en fait de remplacer la proposition (*p*) par la phrase « la porte est fermée », de remplacer la proposition (*q*) par « quelqu'un appuie sur le bouton 3 » et de remplacer la proposition (*r*) par « l'ascenseur part au troisième étage », pour décrire de manière formelle un comportement réel.

En électronique, le même comportement est représenté par deux interrupteurs en série, le premier correspondant à la détection de la fermeture de la porte et le second à la commande du bouton étage 3.

La figure 1.1 illustre ces trois représentations.

La variable logique *p* caractérise la proposition (*p*). Elle peut prendre les deux états vrai ou faux : la porte est fermée ou ouverte. En électronique, on parle d'états haut et bas ou d'états « 1 » et « 0 ». Il en est de même pour les variables *q* et *r*. Les variables logiques *p* et *q* sont des entrées du système alors que la variable *r* est une sortie. La table 1.1.d, appelée **table de vérité** du système, est donnée pour les deux représentations des états logiques. Contrairement au ET logique, le « et » de la langue est insuffisant car les cas « porte ouverte » et « bouton non actionné » ne sont pas pris en compte. Le « et » de la langue est incomplet pour décrire le comportement du système. Par contre, en logique électronique, toutes les combinaisons possibles des états des variables d'entrée sont envisagées et l'état de la variable de sortie est donné dans chaque cas.

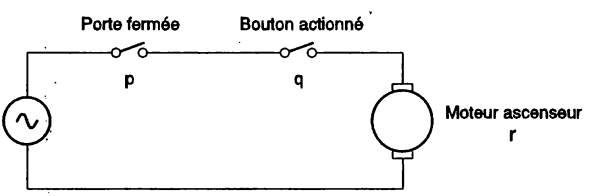
Le schéma 1.1.c est une simplification du système réel. C'est pourquoi il est nécessaire de décrire le schéma 1.2 correspondant à une situation réelle. La détection de la fermeture de la porte, à l'aide d'un contact électrique ou d'une barrière lumineuse, est convertie en un niveau électrique, 0 ou 5 V. Il en est de même pour convertir la poussée sur le bouton de l'étage 3. La sortie du système logique correspond à la variable *r*. Elle est également matérialisée par un niveau électrique, 0 ou 5 V. Ce niveau permet ensuite, à l'aide de circuits d'interface convenablement choisis, d'actionner le moteur de l'ascenseur. Les valeurs 0 et 5 V ne sont que des valeurs données en exemple et dans la pratique, les circuits logiques fonctionnent selon les familles considérées avec des valeurs très variables pour les deux niveaux logiques.




*1.1.a*

| | |
|---|---|
| (p) : porte fermée<br>(q) : quelqu'un appuie bouton 3<br>(r) : l'ascenseur part à l'étage 3 | (r) = (p) et (q) |

*1.1.b*




*1.1.c*

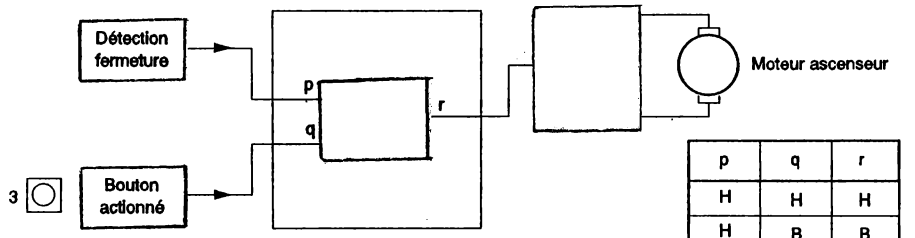
| p | q | r |
|---|---|---|
| V | V | V |
| V | F | F |
| F | V | F |
| F | F | F |

| p | q | r |
|---|---|---|
| 1 | 1 | 1 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 0 | 0 | 0 |

| p | q | r |
|---|---|---|
| H | H | H |
| H | B | B |
| B | H | B |
| B | B | B |

*1.1.d*

Fig. 1.1
Logique et états logiques.




| p | q | r |
|---|---|---|
| H | H | H |
| H | B | B |
| B | H | B |
| B | B | B |

Fig. 1.2
Système logique.

La figure 1.3 représente un système logique combinatoire de la manière la plus générale. Les entrées sont en nombre *n* et les sorties sont en nombre *m*. Chaque sortie est une fonction logique des entrées et d'elles seules. Une sortie ne dépend pas d'une autre sortie par exemple. Pour décrire le système, il faut donc considérer toutes les combinaisons possibles des entrées. Le cas de trois entrées est traité figure 1.4. De manière générale, il y a donc $2^n$ cas à considérer pour *n* entrées. Dans certains cas, les valeurs des sorties peuvent ne pas être définies. L'état est alors vrai ou faux indépendamment des entrées. Il est alors dit **indéfini** et noté par le symbole X. Il peut en être de même pour une entrée. La table de vérité de la figure 1.4 fait apparaître quelques états indéfinis. La table de vérité est également appelée **table de Karnaugh** du nom de son inventeur.

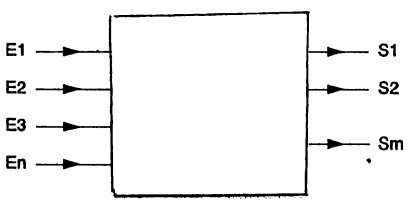
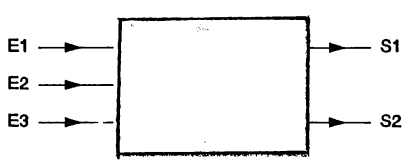



Fig. 1.3
Logique combinatoire dans le cas général.




| E1 | E2 | E3 | S1 | S2 |
|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 1 | X |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | X | 1 |
| 1 | 1 | 0 | X | 1 |
| 1 | 1 | 1 | 0 | 1 |

Fig. 1.4
Table de vérité d'un système combinatoire (exemple).

## Portes logiques

L'exemple de l'ascenseur a permis de mettre en équation logique un problème donné dans un cas très simple. Nous supposerons maintenant que le problème à traiter est connu de manière formelle, c'est-à-dire que les sorties et les entrées sont reliées par une table de vérité donnée et connue. Dans certains cas, il n'est pas toujours facile d'établir cette table et les exemples du dernier paragraphe illustreront les difficultés. La question qui se pose alors est de réaliser le système électronique correspondant à la table de vérité.

Nous admettrons que toute fonction logique combinatoire peut se réaliser à partir de deux fonctions élémentaires, le **ET** logique et le **OU** logique. La première étape est donc de décrire ces deux fonctions.

| E1 | E2 | S |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

| E1 | E2 | S |
|---|---|---|
| F | F | F |
| F | V | F |
| V | F | F |
| V | V | V |

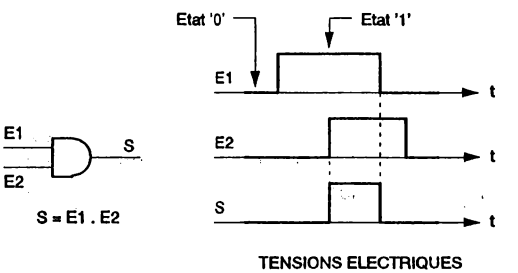


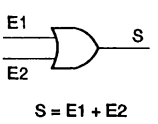

Fig. 1.5
Porte ET.

La figure 1.5 représente la table de vérité de la fonction **ET**, le symbole électronique et les tensions électriques en entrée et en sortie. Le ET logique entre $E_1$ et $E_2$ est noté $E_1 \cdot E_2$, c'est-à-dire comme la multiplication pour des raisons qui seront données par la suite. La sortie est à l'état haut quand les deux entrées sont simultanément à l'état haut. On suppose que le niveau électrique correspondant à l'état « 0 » est la tension nulle et que le niveau électrique correspondant à l'état « 1 » est une tension positive. La figure 1.5 montre alors l'évolution dans le temps de la tension de sortie quand les états d'entrée évoluent. La fonction logique étudiée précédemment pour la mise en marche de l'ascenseur est une fonction ET. Il faut remarquer que les valeurs électriques correspondant aux deux états logiques ne sont pas nécessairement 0 V et une valeur positive. Certaines familles logiques introduisent par exemple deux valeurs négatives de la tension. La fonction ET peut se généraliser à un nombre quelconque d'entrées. Dans ce cas, elle prend la valeur « 1 » quand toutes les entrées prennent simultanément la valeur « 1 ».

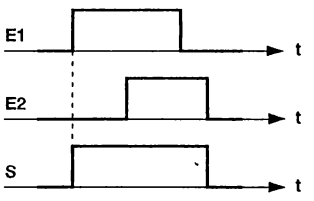
La figure 1.6 représente la deuxième fonction logique de base, à savoir la fonction **OU**. La fonction est notée $E_1 + E_2$, c'est-à-dire comme l'addition, pour des raisons qui seront données par la suite. La sortie est à l'état haut quand l'une au moins des entrées est à l'état haut, et à l'état bas quand les deux entrées sont simultanément à l'état bas. Les tensions en entrée et en sortie sont représentées en fonction du temps. Les mêmes remarques que précédemment s'appliquent quant à la valeur des tensions. La fonction se généralise facilement à un nombre quelconque d'entrées. Elle prend la valeur « 1 » quand l'une au moins des entrées prend la valeur « 1 ».

| E1 | E2 | S |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

| E1 | E2 | S |
|---|---|---|
| F | F | F |
| F | V | V |
| V | F | V |
| V | V | V |




S = E1 + E2



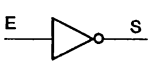

TENSIONS ELECTRIQUES

Fig. 1.6
Porte OU.

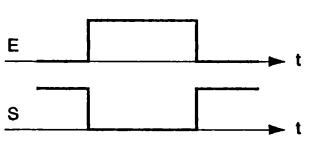
La figure 1.7 représente une fonction logique utilisée fréquemment : l'inverseur. Notée $\overline{E}$, elle inverse le niveau logique d'entrée. Les signaux électriques correspondants sont en opposition.

| E | S |
|---|---|
| 0 | 1 |
| 1 | 0 |

| E | S |
|---|---|
| F | V |
| V | F |




$S = \overline{E}$




TENSIONS ELECTRIQUES

Fig. 1.7
Inverseur.

Voyons maintenant comment un système logique combinatoire quelconque peut se représenter par une combinaison de portes ET et de portes OU. Pour cela, il faut partir de la table de vérité ou table de Karnaugh du système. La figure 1.8 donne un exemple

qu'il est facile de généraliser. La table de vérité est donc supposée connue. Deux sorties $S_1$ et $S_2$ dépendent de trois entrées $E_1$, $E_2$ et $E_3$. La sortie $S_1$ peut s'exprimer en fonction des entrées en remarquant qu'elle prend la valeur « 1 » pour l'une des trois combinaisons grisées sur le tableau. Elle s'exprime donc comme le OU de ces trois combinaisons. La première combinaison correspond à un état « 0 » pour $E_1$, un état « 1 » pour $E_2$ et un état « 1 » pour $E_3$. Cette combinaison correspond donc à la fonction $\bar{E}_1 E_2 E_3$. Le lecteur pourra écrire les autres termes de la même manière. En définitive, la fonction $S_1$ s'écrit :

$$S_1 = \bar{E}_1 E_2 E_3 + E_1 \bar{E}_2 E_3 + E_1 E_2 \bar{E}_3$$

De la même manière, la sortie $S_2$ s'écrit :

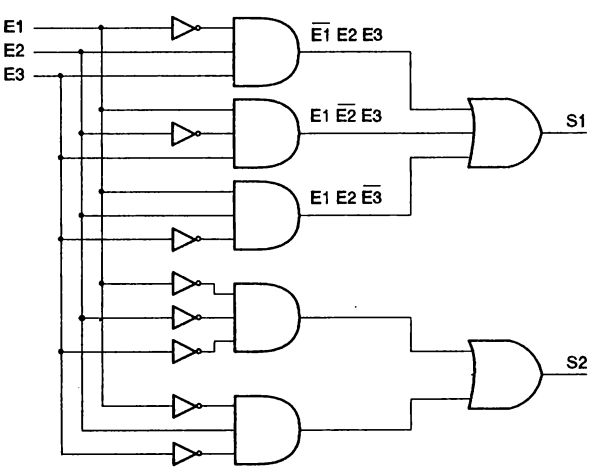
$$S_2 = \bar{E}_1 \bar{E}_2 \bar{E}_3 + \bar{E}_1 E_2 \bar{E}_3$$

Il est alors élémentaire de passer au schéma électrique réalisant la fonction.

Fig. 1.8
Synthèse d'un système logique.

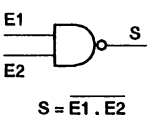
| E1 | E2 | E3 | S1 | S2 |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 0 |

S1 = $\bar{E1}$ E2 E3 + E1 $\bar{E2}$ E3 + E1 E2 $\bar{E3}$
S2 = $\bar{E1}$ $\bar{E2}$ $\bar{E3}$ + $\bar{E1}$ E2 $\bar{E3}$



D'autres portes sont très utilisées dans les systèmes logiques. Ce sont les portes NAND, NOR et le OU exclusif. Elles ne sont pas indispensables à la synthèse des fonctions logiques. Les fonctions ET et OU suffisent pour réaliser une fonction quelconque. Elles sont cependant réalisées par des combinaisons simples de transistors et sont donc fournies par les fabricants de circuits. Les figures 1.9.a et b illustrent ces deux fonctions. Elles sont simplement obtenues en inversant les fonctions ET et OU.

| E1 | E2 | S |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |



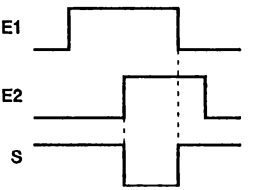
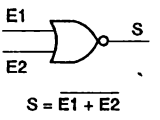




Fig. 1.9.a
Porte NAND.

| E1 | E2 | S |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |



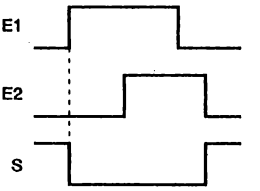
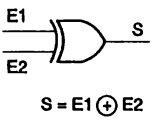




Fig. 1.9.b
Porte NOR.

La fonction OU exclusif est obtenue par une légère modification de la fonction OU. Quand les deux entrées sont simultanément à l'état « 1 », alors la sortie est à l'état « 0 ». La figure 1.10 représente cette fonction d'usage courant en arithmétique binaire.

| E1 | E2 | S |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |



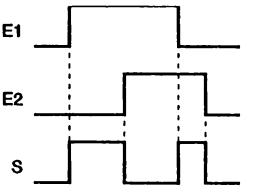




Fig. 1.10
Porte ou exclusif.

De nouveaux symboles sont apparus et sont maintenant normalisés. Ils sont représentés figure 1.11. La correspondance entre les anciens symboles et les nouveaux est également établie figure 1.11. Les deux représentations sont encore utilisées et l'ancienne représentation sera assez souvent pratiquée dans la suite de l'ouvrage.

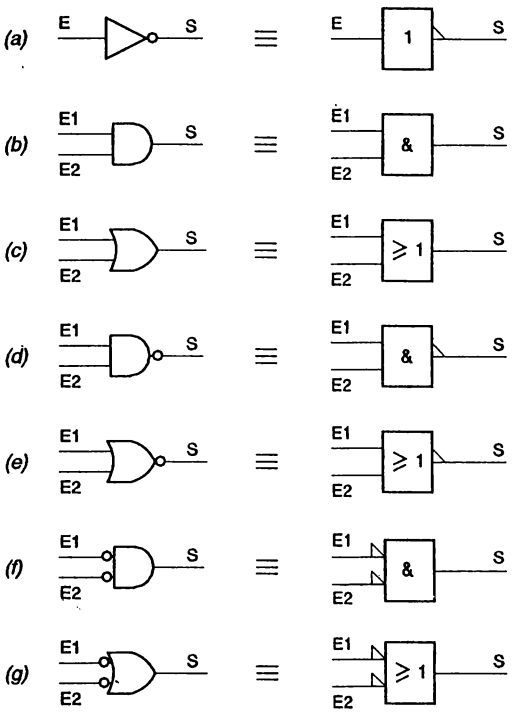



Fig. 1.11
Anciens et nouveaux symboles.

L'inversion est symbolisée par le triangle. La représentation 1.11.f est souvent difficile à assimiler par le débutant. Elle signifie que la fonction représentée est réalisée pour les variables $\overline{E_1}$ et $\overline{E_2}$. La sortie S s'écrit donc :

$$S = \overline{E_1} \cdot \overline{E_2}$$

La formule se transforme alors, comme il sera vu dans le paragraphe suivant :

$$S = \overline{E_1 + E_2}$$

De même, la sortie du circuit 1.11.g s'écrit :

$$S = \overline{E_1} + \overline{E_2}$$

soit :

$$S = \overline{E_1 \cdot E_2}$$

On peut également dire que la fonction est réalisée pour les états bas des entrées.

## Algèbre de Boole

Le paragraphe précédent a montré que toute fonction logique peut s'exprimer sous forme de combinaisons de ET et de OU. Il est maintenant indispensable de donner les propriétés de ces combinaisons dans le but de les simplifier au maximum avant de réaliser les circuits correspondants. L'ensemble de ces techniques constitue **l'algèbre de Boole**, du nom du mathématicien né en 1854 et inventeur du concept.

Les variables sont donc des variables logiques. Elles prennent deux valeurs possibles, « 0 » et « 1 ». Les opérations considérées sont le ET logique noté comme la multiplication et le OU logique noté comme l'addition. Ces notations sont justifiées dans la mesure où les règles de calcul de l'algèbre de Boole sont presque identiques à celles de l'algèbre classique à condition de remplacer le ET par la multiplication et le OU par l'addition. Le but de ce paragraphe n'est pas de démontrer ces règles mais d'apprendre à les utiliser le plus simplement possible. Les démonstrations sont, au demeurant, fort simples. Il suffit de construire les tables de vérité des deux membres et de constater leur équivalence.

### 1 Règles générales

- *Commutativité*

  $AB = BA$
  $A + B = B + A$

- *Associativité*

  $A(BC) = (AB)C$
  $A + (B + C) = (A + B) + C$

- *Distributivité*

  $A(B + C) = AB + AC$

- *Lois spéciales*

  $A + AB = A$
  $A(A + B) = A$

- *Lois de De Morgan*

  $\overline{A+B} = \overline{A}\,\overline{B}$
  $\overline{AB} = \overline{A} + \overline{B}$

### 2 Règles particulières

- *Fonction ET*

  0 0 = 0
  0 1 = 0
  1 0 = 0
  1 1 = 1

$A\,0 = 0$
$0\,A = 0$
$A\,1 = A$
$1\,A = A$
$A\,A = A$
$A\,\overline{A} = 0$

- *Fonction OU*

$0 + 0 = 0$
$0 + 1 = 1$
$1 + 0 = 1$
$1 + 1 = 1$
$A + 0 = A$
$0 + A = A$
$A + 1 = 1$
$1 + A = 1$
$A + A = A$
$A + \overline{A} = 1$

- *Complément*

$\overline{0} = 1$
$\overline{1} = 0$
$\overline{\overline{A}} = A$

## 3 Simplification des expressions logiques

Toutes ces règles se démontrent à l'aide des tables de vérité. Elles sont toutes assez naturelles avec les notations employées à l'exception des lois de De Morgan et des additions du type 1 + 1 = 1. Voyons maintenant comment simplifier une expression booléenne.

L'expression considérée s'écrit :

$$E = B\overline{C}\,\overline{D} + \overline{A}BD + ABD + BC\overline{D}$$

Un regroupement des termes conduit à écrire :

$$E = (\overline{A} + A)BD + B\overline{D}(C + \overline{C})$$

soit :

$$E = BD + B\overline{D}$$

et finalement :

$$E = B$$

La méthode consiste donc à faire apparaître des termes de la forme $A + \overline{A}$ dans l'expression. Elle peut se programmer et a donné lieu à l'algorithme dit de **Quine-Mc Cluskey**.

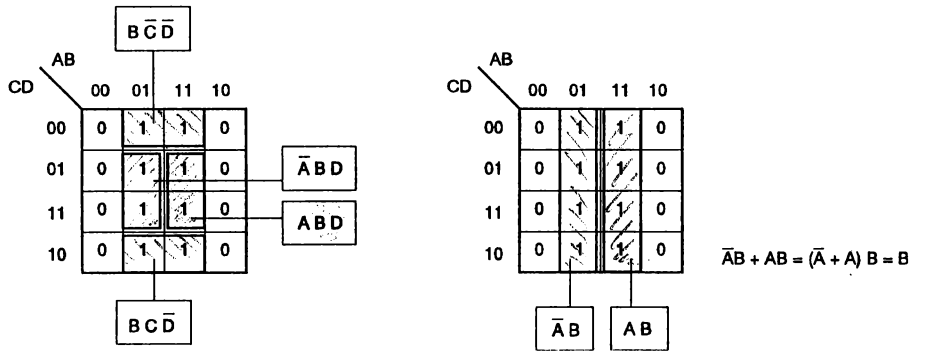
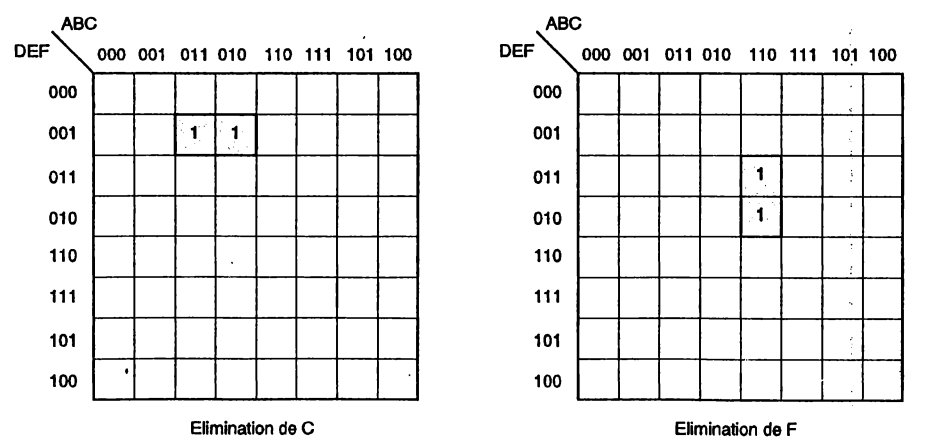
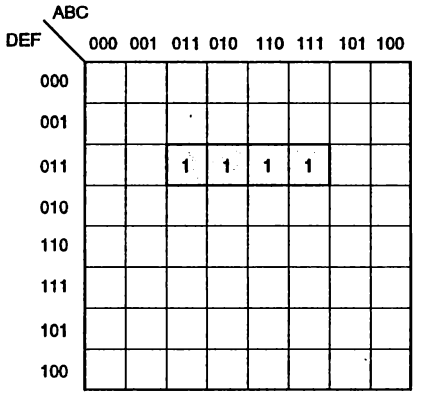



Fig. 1.12
Simplification d'une table de Karnaugh.

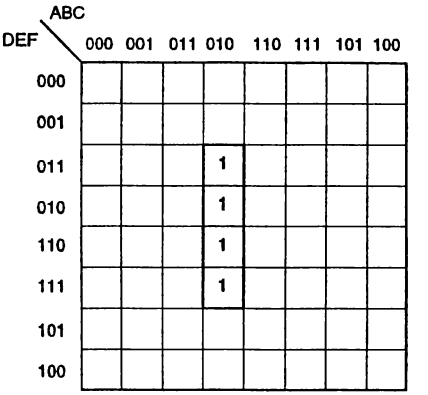
L'autre méthode consiste à simplifier l'expression en raisonnant sur la table de Karnaugh. Reprenons la même expression logique. Elle est représentée graphiquement figure 1.12. La table représente les valeurs de la fonction *E* pour toutes les valeurs possibles des variables *A*, *B*, *C* et *D*. En abscisse, sont portées les valeurs des variables *A* et *B*, et en ordonnée celles de *C* et *D*. La table pourrait être présentée d'une autre manière, par exemple en faisant figurer en abscisse les trois variables *A*, *B* et *C*. Il y aurait alors 8 cases. Les valeurs possibles des variables sont indiquées en tenant compte de l'ordre dans l'expression logique. La valeur 01 pour *AB* veut dire que *A* a la valeur 0 et *B* la valeur 1. Les quatre expressions logiques constituant *E* sont représentées sur le tableau. On constate alors que quatre « 1 » sont adjacents dans la colonne « 01 ». La fonction prend donc la valeur 1 pour $A = 0$ et $B = 1$ indépendamment de *C* et *D*, ce qui donne le terme $\bar{A}B$. De même, on obtient à partir de l'autre colonne de 1 le terme *AB*. En définitive, on obtient $\bar{A}B + AB$ soit *B*.

Fig. 1.13
Regroupements par 2.

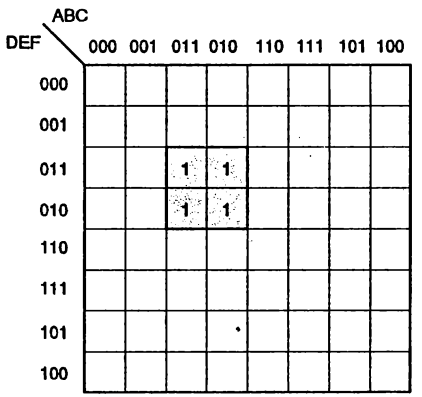





Elimination de AC




Elimination de DF




Elimination de CF

Fig. 1.14
Regroupements par 4.

Les figures 1.13 et 1.14 représentent de manière générale comment des regroupements par 2 ou par 4 peuvent simplifier une expression logique. Des regroupements encore plus importants pourraient être envisagés, par 8 ou plus. Ils sont cependant moins fréquents dans les cas réels. L'exemple suivant permettra de mettre ces techniques en application :

$$E = \overline{A}B\overline{C} + AB\overline{C} + A\overline{B}\overline{C} + \overline{A}\overline{B}C + \overline{A}BC + ABC$$

La table de Karnaugh est représentée figure 1.15. Les six groupes correspondent à six « 1 » dans le tableau. Après simplification sur la table de Karnaugh, on obtient une expression qui peut encore se simplifier comme il est indiqué figure 1.15. Les techniques graphiques et algébriques peuvent donc se combiner. Il ne faut pas oublier la possibilité de regrouper les « 1 » placés en bordure comme l'indique la figure 1.16.

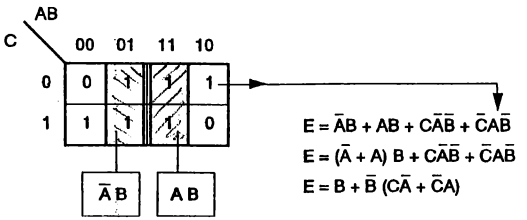



Fig. 1.15
Exemple de simplification.

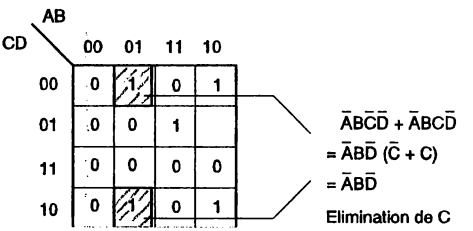



Fig. 1.16
Regroupement sur les bords.

# Représentations binaire, hexadécimale et ASCII

Après avoir étudié les propriétés des variables logiques et les méthodes de simplification, il est maintenant possible de montrer comment on peut représenter des chiffres et des nombres à l'aide de symboles binaires. La représentation binaire des nombres est très classique et donne lieu à l'arithmétique binaire qui est tout à fait semblable à l'arithmétique décimale. L'intérêt de l'arithmétique binaire est, comme il sera vu par la suite, de se prêter facilement à la réalisation électronique des opérations principales.

## 1 Représentation binaire des nombres

De la même manière que l'écriture décimale d'un nombre est sa décomposition selon les puissances de dix, la représentation binaire est sa décomposition selon les puissances de deux. Par exemple, 287 s'écrira :

$$287 = 256 + 16 + 8 + 4 + 2 + 1$$

soit en exprimant les puissances de deux :

$$287 = 1\times2^8 + 0\times2^7 + 0\times2^6 + 0\times2^5 + 1\times2^4 + 1\times2^3 + 1\times2^2 + 1\times2^1 + 1\times2^0$$

L'écriture binaire est donc :

$$287 = 100011111$$

Les coefficients des puissances de deux prennent les valeurs 1 ou 0 et sont appelés **bits**. Le bit le plus à gauche est appelé **bit le plus significatif**. Le bit le plus à droite est le **bit le moins significatif**. Les termes anglais sont **MSB** et **LSB**. Pour obtenir l'écriture binaire d'un nombre à partir de l'écriture décimale, il suffit d'effectuer une suite de divisions par 2 et de noter les restes à chaque division. La série des restes est l'écriture binaire du nombre en commençant par le LSB.

Les nombres fractionnaires s'exprimeront en fonction des puissances négatives de deux. Par exemple, le nombre binaire 100011111.11 s'écrira :

$$100011111.11 = 287 + 2^{-1} + 2^{-2}$$

Les opérations s'effectuent en binaire comme en décimal. Les quelques opérations ci-dessous permettent de s'en assurer. Pour l'addition, par exemple, il suffit de poser l'opération et de tenir compte des reports. L'addition 287 + 33 s'écrit alors :

```
  1 0 0 0 1 1 1 1 1
+       1 0 0 0 0 1
-----------------------
  1 0 1 0 0 0 0 0 0
```

On obtient bien évidemment :

$$2^8 + 2^4 = 64 + 256 = 320$$

La soustraction s'effectue de la même manière. Il faut alors tenir compte des retenues. Le calcul de 320 moins 33 s'écrit :

```
  1 0 1 0 0 0 0 0 0
-       1 0 0 0 0 1
-----------------------
  1 0 0 0 1 1 1 1 1
```

On obtient bien évidemment 287.

La multiplication se pose comme en décimal et se calcule en tenant compte de la table de multiplication binaire. Le produit 13 fois 5 s'écrit par exemple :

```
        1 1 0 1
×         1 0 1
-----------------
        1 1 0 1
      0 0 0 0
    1 1 0 1
-----------------
  1 0 0 0 0 0 1
```

Le résultat est 65. La multiplication binaire se réalise donc par une suite de décalages à gauche et une addition finale.

La division binaire pourrait se présenter également selon le schéma de la division décimale.

| Décimal | Binaire | Hexa-décimal |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 1 | 1 |
| 2 | 10 | 2 |
| 3 | 11 | 3 |
| 4 | 100 | 4 |
| 5 | 101 | 5 |
| 6 | 110 | 6 |
| 7 | 111 | 7 |
| 8 | 1000 | 8 |
| 9 | 1001 | 9 |
| 10 | 1010 | A |
| 11 | 1011 | B |
| 12 | 1100 | C |
| 13 | 1101 | D |
| 14 | 1110 | E |
| 15 | 1111 | F |
| 16 | 10000 | 10 |
| 17 | 10001 | 11 |
| 18 | 10010 | 12 |
| 19 | 10011 | 13 |
| 20 | 10100 | 14 |

## 2 Représentation hexadécimale

La représentation hexadécimale est une représentation en base 16. Son utilisation est fréquente en informatique. Elle sert en particulier à définir les adresses. Le tableau ci-contre donne les représentations décimales, binaires et hexadécimales des 20 premiers nombres.

La suite s'en déduit sans difficulté. Si nous voulons transcrire un nombre binaire en hexadécimal, il suffit de grouper les bits par quatre et de remplacer chaque groupe par le nombre hexadécimal correspondant. Par exemple, on écrira :

1000001 = 100 0001 = 41

100011111 = 1 0001 1111 = 11F

De même, il est facile de passer de la représentation hexadécimale à la représentation binaire en remplaçant chaque symbole hexadécimal par les quatre bits correspondants.

11F = 1 0001 1111

## 3 Le code ASCII

Contrairement aux codes précédents qui étaient limités au codage des nombres, le code ASCII a été défini pour coder les symboles du clavier d'une machine à écrire ou d'un clavier informatique. Les nombres sont codés mais aussi les lettres et les symboles de l'imprimerie ainsi qu'un certain nombre de caractères de contrôle comme le retour chariot ou le saut de page. Le code ASCII *(American Standard Code for Information Interchange)* est d'un usage universel et tous les fichiers texte informatiques adoptent ce standard.

Chaque symbole est codé sur 8 bits. Le premier bit est un bit de contrôle qui ne porte pas d'information. Il est appelé **bit de parité**. Sa valeur est « 0 » quand il y a un nombre pair de « 1 » dans les 7 bits restants du symbole. Sa valeur est « 1 » quand le nombre de « 1 » est impair. Pour l'obtenir, on peut effectuer la somme binaire des 7 symboles. Le bit de parité est alors le bit le moins significatif de la somme. Il est ainsi possible de contrôler après une transmission de données si le bit de parité de chaque symbole a une valeur normale ou non. Sa valeur doit être identique à l'émission et à la réception. Si la valeur du bit de parité n'est pas conforme au symbole, il est possible de demander une nouvelle émission. Le bit de parité est le premier bit du symbole ASCII. Le tableau 1.17 donne une liste partielle de symboles ASCII.

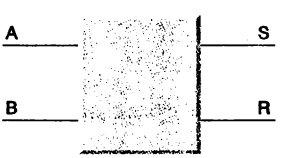
| Caractère | ASCII à 7 éléments | Caractère | ASCII à 7 éléments |
|---|---|---|---|
| A | 100 0001 | Y | 101 1001 |
| B | 100 0010 | Z | 101 1010 |
| C | 100 0011 | 0 | 011 0000 |
| D | 100 0100 | 1 | 011 0001 |
| E | 100 0101 | 2 | 011 0010 |
| F | 100 0110 | 3 | 011 0011 |
| G | 100 0111 | 4 | 011 0100 |
| H | 100 1000 | 5 | 011 0101 |
| I | 100 1001 | 6 | 011 0110 |
| J | 100 1010 | 7 | 011 0111 |
| K | 100 1011 | 8 | 011 1000 |
| L | 100 1100 | 9 | 011 1001 |
| M | 100 1101 | blanc | 010 0000 |
| N | 100 1110 | . | 010 1110 |
| O | 100 1111 | ( | 010 1000 |
| P | 101 0000 | + | 010 1011 |
| Q | 101 0001 | $ | 010 0100 |
| R | 101 0010 | * | 010 1010 |
| S | 101 0011 | ) | 010 1001 |
| T | 101 0100 | _ | 010 1101 |
| U | 101 0101 | / | 010 1111 |
| V | 101 0110 | , | 010 1100 |
| W | 101 0111 | = | 011 1101 |
| X | 101 1000 | (Retour) | 000 1101 |
| | | (Interligne) | 000 1010 |

Fig. 1.17
Tableau partiel du code ASCII.

## Arithmétique binaire

Les opérations de l'arithmétique binaire peuvent se réaliser par des associations de portes logiques. Dans un premier temps, l'addition est décrite. La figure 1.18 représente un **additionneur** 1 bit. Les deux entrées *A* et *B* donnent une sortie *S* et éventuellement un report *R*. La table de vérité s'obtient à partir des règles de l'addition :

0 + 0 = 0
1 + 0 = 1
0 + 1 = 1
1 + 1 = 0 (dans ce cas, il y a un report)




| A | B | S | R |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 |

Fig. 1.18
Additionneur 1 bit.

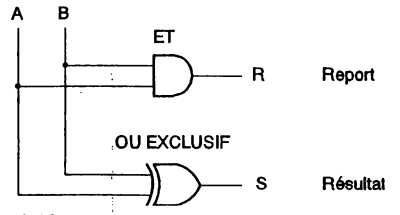
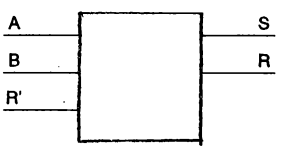



Fig. 1.19
Additionneur 1 bit, réalisation.

La table de vérité donne alors lieu à la réalisation de la figure 1.19. Les fonctions sont réalisées par un OU exclusif et par une porte ET.

Il est maintenant possible de généraliser à un nombre de bits quelconque. Le cas de 4 bits est pris comme exemple. La présence éventuelle du report conduit à définir un additionneur un peu plus complexe. Cet additionneur élémentaire est décrit figure 1.20. Il correspond à l'opération effectuée sur deux bits des deux nombres à additionner. Le report en entrée correspond au report éventuel des deux bits de poids plus faibles. Ce circuit est appelé **additionneur complet**. La figure 1.21 représente l'additionneur 4 bits complet. Il est donc réalisé à l'aide d'un additionneur simple et de trois additionneurs un bit complets. Les bits *A*0 et *B*0 sont les bits de poids faible. Les bits *A*3 et *B*3 sont les bits de poids fort. Le report résultant de l'addition de deux bits est appliqué en entrée de l'additionneur de poids plus élevé. Le report du dernier additionneur est le bit de poids le plus élevé du résultat, soit $S_4$. La réalisation de l'additionneur un bit complet à partir de portes est proposée au lecteur à titre d'exercice.




| A | B | R' | S | R |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

Fig. 1.20
Additionneur avec report.

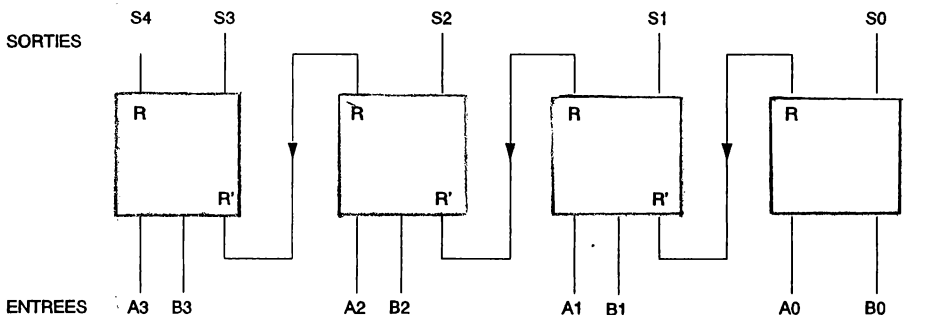



Fig. 1.21
Additionneur 4 bits.

La soustraction binaire se réalise de la même manière à partir d'un soustracteur un bit élémentaire. Le report a un équivalent, la retenue.

La multiplication se réalise par une addition des différents nombres obtenus après décalage d'un bit et multiplication par 0 ou 1. La fonction de décalage peut se réaliser par un circuit spécial comme il sera vu par la suite. La multiplication peut donc se réaliser à partir de circuits logiques. Le circuit ainsi formé est appelé multiplicateur câblé.

Ce paragraphe a donc montré comment il est possible d'effectuer des opérations simples en arithmétique binaire à l'aide de portes logiques.

Des opérations plus complexes peuvent également être réalisées comme par exemple l'addition, la soustraction ou la multiplication de nombres flottants, c'est-à-dire comprenant des chiffres derrière une virgule.

Dans de nombreux cas cependant, les calculateurs électroniques ne réalisent que les opérations binaires les plus simples : addition et soustraction. Les opérations plus complexes se réalisent à partir des opérations simples par des programmes de calcul adaptés.

## Exemples de fonctions combinatoires

Quatre exemples illustrent ce chapitre. Le premier est un multiplexeur logique, le second une commande d'afficheur, le troisième un circuit logique programmable, le dernier la description d'une mémoire de type ROM.

### 1 Démultiplexeur et multiplexeur logique

La fonction du **démultiplexeur** est d'envoyer les bits d'un message vers une voie choisie parmi *n*.

Les applications de ce circuit sont multiples. Il est ainsi possible d'envoyer les données d'un ordinateur vers un périphérique donné, imprimante ou écran. Dans l'exemple, l'information est supposée codée sur 8 bits et 3 voies de sortie sont possibles. La voie choisie est codée en binaire.

Le schéma synoptique est représenté figure 1.22. Les bits de sélection indiquent la voie choisie. La figure 1.22 montre comment la fonction peut être réalisée pour chacun des 8 bits à l'aide de 3 portes ET. Quand le niveau « 1 » est appliqué en entrée d'une porte, le bit présent sur l'autre entrée se retrouve en sortie. Si le niveau « 0 » est appliqué, la sortie est toujours au niveau « 0 ». Les bits de sélection seront alors positionnés selon la table de vérité indiquée figure 1.23. On en déduit les équations logiques des variables $B_1$, $B_2$ et $B_3$. La réalisation est immédiate à partir de portes ET.

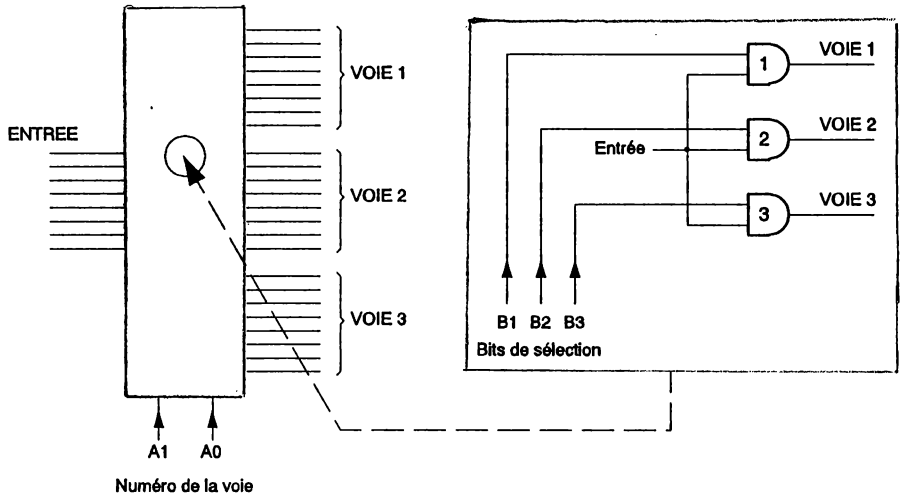



Fig. 1.22
Démultiplexeur logique.

| | A1 | A0 | B1 | B2 | B3 |
|---|---|---|---|---|---|
| Voie 1 | 0 | 1 | 1 | 0 | 0 |
| Voie 2 | 1 | 0 | 0 | 1 | 0 |
| Voie 3 | 1 | 1 | 0 | 0 | 1 |

$B1 = \overline{A1}\ A0$
$B2 = A1\ \overline{A0}$
$B3 = A1\ A0$

Fig. 1.23
Table de vérité.

La figure 1.24 montre la réalisation de l'ensemble. On notera, comme c'est souvent le cas en logique, la répétition du même motif.

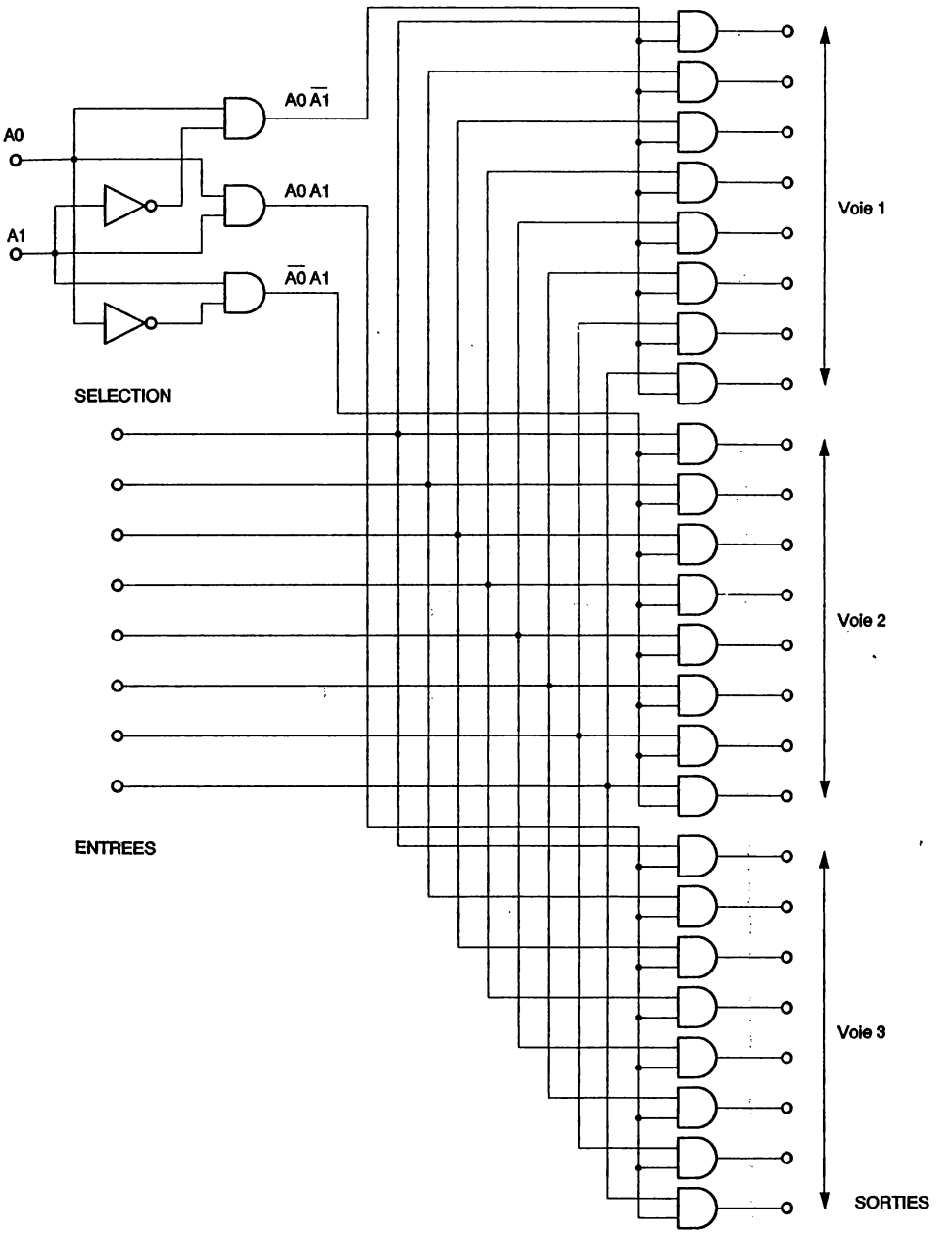



Fig. 1.24
Démultiplexeur 8 bits, 3 voies.

La fonction inverse peut également être réalisée. Le schéma 1.25 représente ce circuit, appelé **multiplexeur**, qui permet de choisir une voie parmi *n*. Les applications sont nombreuses : sélection d'une voie de mesure avant traitement, transmission de données en mode série, etc.

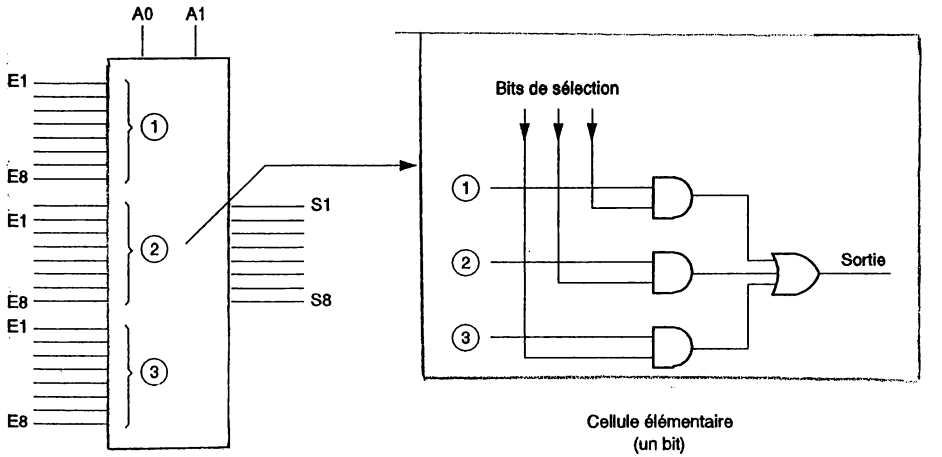
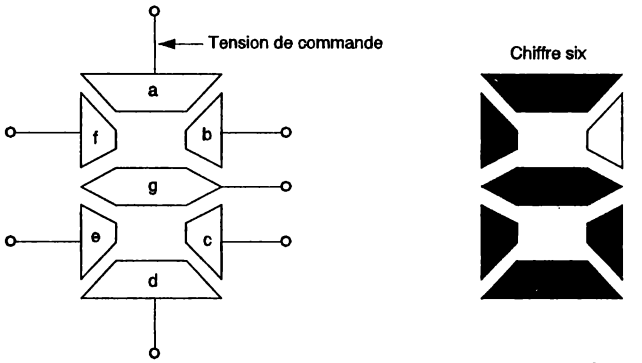



Fig. 1.25
Multiplexeur logique.

## Commande d'afficheur

Pour afficher un nombre décimal, compris entre 0 et 9, on utilise souvent des afficheurs à segments comme le représente la figure 1.26. Les applications sont multiples : affichage de l'heure, indication de fonction, résultat de mesure, etc. Les segments sont généralement des cristaux liquides. Quand une tension est appliquée, le segment se noircit. En fonction des segments commandés, il est possible de faire apparaître les dix chiffres du système décimal. Le chiffre six est donné en exemple mais le lecteur pourra construire les autres chiffres en fonction du tableau de la figure 1.26. Le circuit de commande reçoit en entrée les 4 bits du chiffre à afficher et fournit en sortie les commandes adéquates aux segments d'affichage. En appelant *a, b, c, d, e, f* et *g*, les variables logiques correspondant aux tensions de commande des segments et $B_0$, $B_1$, $B_2$ et $B_3$, les bits de codage du chiffre, il est possible d'écrire les équations logiques du circuit. Il est plus simple de partir des tables de Karnaugh pour simplifier au maximum. Certaines combinaisons n'apparaissent pas dans la table 1.26, elles correspondent aux chiffres compris entre 10 et 15. Les sorties peuvent alors prendre un état quelconque. Cela sera sans incidence sur l'affichage puisque ces combinaisons n'apparaissent jamais en entrée. Dans la table, la sortie est alors affectée du symbole X.




| Nombre à afficher | SEGMENTS NOIRCIS | | | | | | | CODAGE BINAIRE DU NOMBRE | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | a | b | c | d | e | f | g | B3 | B2 | B1 | B0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 2 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 |
| 3 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 |
| 4 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 |
| 5 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 |
| 6 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 |
| 7 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 8 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 9 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 |

Fig. 1.26
Afficheur à cristaux liquides.

La figure 1.27 montre la construction de la table de Karnaugh de la fonction *a* et sa simplification. Il est intéressant de grouper le plus de « 1 » possible et donc de profiter de la valeur indéterminée « X ». Cet exemple montre deux groupements par 8, un groupement par quatre et un groupement par deux. On obtient alors l'équation de la figure 1.27. En fait, il est possible de regrouper les termes de manière encore plus astucieuse. Ce regroupement est indiqué figure 1.28. Les termes sont pris sur les bords de la table. Ils n'apparaissent disjoints que par simple convention dans l'ordre de numérotation des variables. Ce regroupement de quatre termes fait apparaître un produit $\overline{B}_2\overline{B}_0$ plus simple que le produit $\overline{B}_1\overline{B}_2\overline{B}_0$. La construction des tables des autres fonctions et leur simplification sont proposées à titre d'exercice. Les résultats sont indiqués ci-après :

$$a = B_2B_0 + B_3 + B_1 + \overline{B_2}\,\overline{B_0}$$
$$b = \overline{B_2} + B_1B_0 + \overline{B_1}\,\overline{B_0}$$
$$c = B_2 + B_0 + \overline{B_1}$$
$$d = \overline{B_2}\,\overline{B_0} + \overline{B_2}B_1 + B_1\overline{B_0} + B_2\overline{B_1}B_0$$

$$e = \overline{B_2}\,\overline{B_0} + B_1\overline{B_0}$$
$$f = B_3 + \overline{B_1}\,\overline{B_0} + B_2\overline{B_1} + B_2\overline{B_0}$$
$$g = B_3 + B_2\overline{B_1} + \overline{B_2}B_1 + B_1\overline{B_0}$$

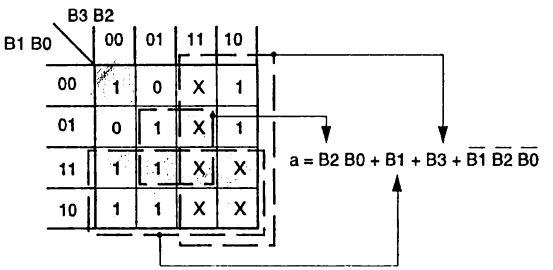



Fig. 1.27
Équation logique de a.

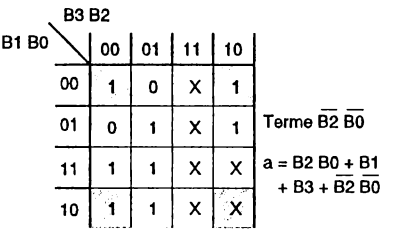



Fig. 1.28
Regroupement moins évident.

Il est alors possible de réaliser le circuit comme l'indique la figure 1.29.

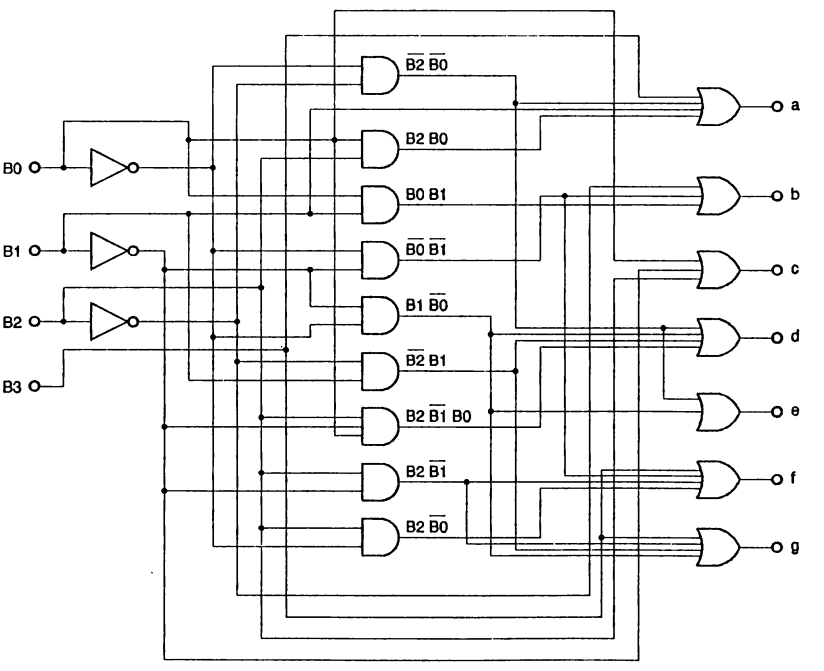


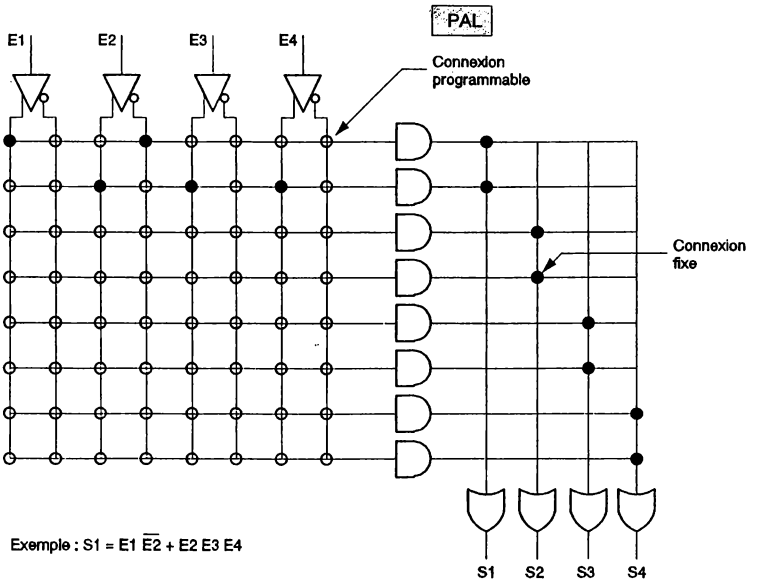

Fig. 1.29
Commande d'afficheur.

## 3 Circuits programmables

Il ne s'agit pas d'une application proprement dite mais plutôt d'une technologie particulière pour réaliser des fonctions logiques. Les exemples précédents ont montré qu'il était possible de synthétiser une équation logique ou une table de vérité à l'aide de portes. L'industrie électronique fournit des boîtiers comportant un certain nombre de portes (de 4 à 8 en général). Il suffit de les alimenter et de les interconnecter selon les principes décrits précédemment. Cette technique est cependant du domaine du passé et la conception moderne de l'électronique conduit à intégrer au maximum les fonctions dans des circuits spéciaux appelés **circuits programmables**. Ces circuits comportent donc un certain nombre de portes logiques qu'il suffit de connecter pour réaliser la fonction désirée. C'est en fait une miniaturisation des techniques classiques puisque les interconnexions sont réalisées sur le silicium et non pas par des pistes imprimées. L'encombrement est alors réduit et le coût du circuit diminue d'autant.

Deux types de circuit sont représentés : le **PAL** (*programmable array logic*) et le **PLA** (*programmable logic array*). Le circuit se présente sous forme d'une matrice de points de connexions programmables. Les techniques de programmation diffèrent selon les fabricants mais le résultat est toujours le même : le point de connexion est établi. Quand plusieurs points de connexion sont réalisés sur une ligne, la fonction ET entre les entrées concernées est produite. La figure 1.30 représente cette fonction de manière simplifiée pour ne pas trop charger le dessin. Des points de connexions, fixes dans le cas du PAL, permettent de réaliser des OU de ces diverses fonctions. Dans le cas du PLA, ces points de connexions sont également programmables et la souplesse d'utilisation est plus grande. La figure donne deux exemples de fonction réalisables. Les points de connexions créés sont alors noircis sur le schéma. Dans la pratique, les circuits peuvent être beaucoup plus complexes et certaines familles comportent jusqu'à plusieurs dizaines de milliers de portes.

La réalisation d'un circuit programmable fait largement appel aux outils de conception assistée par ordinateur (CAO). Le circuit programmable est placé sur un support relié à un ordinateur de type PC ou à une station de travail. Des logiciels évolués permettent de transformer les équations logiques ou les tables de vérité en un système logique simplifié. Ils permettent ensuite de créer le fichier de programmation qui sert à établir les points de connexions. Dans certaines familles, des fusibles sont grillés en faisant passer un courant suffisant et les points de connexions sont établis. Parmi les fabricants de circuits programmables, on peut citer *XILINX*, *ACTEL*, *ALTERA* et *MAXIM*.



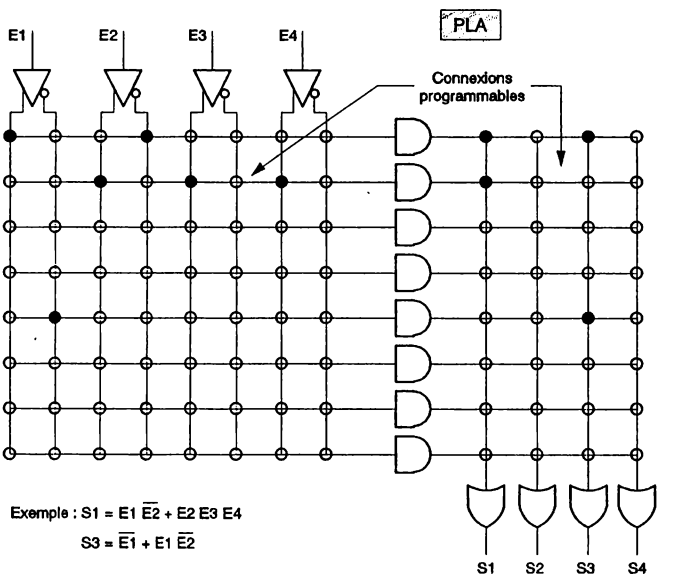




Fig. 1.30
Circuits programmables PAL et PLA.

## 4 Les mémoires ROM

Les mémoires **ROM** (***Read only memory***) font partie de la grande famille des mémoires électroniques. Elles seront à ce titre étudiées dans le chapitre consacré aux familles logiques. Elles peuvent cependant être considérées comme des circuits combinatoires programmables. Une mémoire est un classeur électronique dont chaque case est remplie par un mot binaire donné. En général, les mots sont de 8 bits. Le nombre de cases est la taille de la mémoire. Une mémoire de 1 K comporte par exemple 1 024 cases remplies de mots de 8 bits. La mesure 1 Kbyte est une simplification du chiffre exact 1 024 qui est une puissance de 2 et correspond donc au circuit réel. De même, une mémoire de 1 Mbyte comporte en fait 1 024 fois 1 024 emplacements de 8 bits. Le **byte** désigne un groupe de 8 bits. Les mémoires CMOS actuelles ont des tailles aussi élevées que 16 Mbytes. À chaque adresse correspond un mot de 8 bits. La mémoire ROM réalise une conversion logique entre un mot d'entrée, l'**adresse**, et un mot de sortie, 8 bits en général, qui est le contenu de l'adresse. Certaines mémoires peuvent s'effacer et se programmer à nouveau. Elles sont appelées **EPROM** ou **EEPROM** selon que l'effacement s'effectue par une exposition aux rayons ultraviolets ou par l'application d'un signal électrique.

## ENTRAÎNEMENT

**1** Écrivez en hexadécimal : 100100110101.

**2** Écrivez en binaire 263 et 187. Posez l'addition, la soustraction et la multiplication de ces deux nombres.

**3** Représentez par des équations logiques un va-et-vient électrique.

**4** Reprenez l'exemple de l'ascenseur pour un immeuble de deux étages. Le cas où deux appels sont simultanés doit être envisagé. Dans ce cas, la priorité est donnée au rez-de-chaussée. Réalisez la logique de commande.

**5** Simplifiez l'expression :

$$E = A\overline{BC} + \overline{AC}D + ABD + BCD + B\overline{D}$$

**6** Démontrez les relations : $AB = BA$

$$\overline{A+B} = \overline{A}\,\overline{B}$$

On écrira les tables de vérité de chaque expression.

**7** Réalisez un additionneur 8 bits à partir d'additionneurs 4 bits.

**8** Montrez dans l'exemple de la commande d'afficheur que les variables *b, c, d, e, f* et *g* sont données par les formules de la page 26.

Chapitre 2

# LOGIQUE SÉQUENTIELLE

## SOMMAIRE


- Bascule RS
- Bascule D
- Bascule JK maître-esclave
- Registres à décalage
- Applications des registres à décalage
- Compteurs décompteurs
- Applications des compteurs
- Description des systèmes de logique séquentielle en machines d'états
- Entraînement


*Lorsque l'on parle de logique combinatoire, on s'intéresse à des systèmes où les sorties prennent des états en fonction des états d'entrée et* ***uniquement*** *en fonction de ces états.*
*En logique séquentielle, on introduit une nouvelle variable : le temps.*
*Les états des sorties ne sont plus fonction seulement des états d'entrée mais fonction des états d'entrée et du temps.*
*Dans la pratique, un sous-ensemble logique sera constitué de logique combinatoire, et dans de nombreux cas, de logique séquentielle.*
*Ce chapitre sera consacré à la description des bascules, sous-ensembles élémentaires permettant la réalisation de fonctions plus complexes comme les compteurs et les registres à décalage.*
*Les bascules sont réalisées à partir de portes AND, NAND, OR et NOR.*
*En conséquence, la lecture du chapitre précédent est un préalable obligatoire.*

## Bascule RS

Le circuit élémentaire de la logique séquentielle est la bascule RS dont le schéma est donné à la figure 2.1.

Cette bascule peut être réalisée soit à partir de portes NAND soit à partir de portes NOR.

Le terme RS signifie Reset-Set.

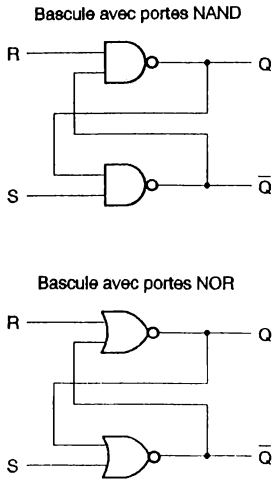



Fig. 2.1
Bascule RS.

## 1 Bascule RS avec portes NAND

Pour cette bascule, la table de vérité est la suivante :

| $R$ | $S$ | $Q_{n(t)}$ | $\overline{Q_{n(t)}}$ |
|---|---|---|---|
| 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 1 | $Q_{n-1(t)}$ | $\overline{Q_{n-1(t)}}$ |
| 0 | 0 | indéterminé | indéterminé |

L'examen de la troisième ligne, où $R = S = 1$, décrit le fonctionnement de la bascule. Dans ce cas, les sorties ne changent pas d'état et conservent l'état précédent. Tout se passe comme si le circuit avait **mémorisé** l'état précédent : le passage de $R$ et $S$ à 1.

Pour stocker un 0 sur $Q$, il suffit d'appliquer $R = \overline{S} = 1$ et pour stocker 1, $R = \overline{S} = 0$.

Avec $R = S = 1$, l'information est conservée sur $Q$.

La bascule RS constitue donc une mémoire élémentaire d'un bit. Pour $R = S = 0$, les états des sorties sont indéterminés. Cette configuration doit donc être interdite en entrée.

## 2 Bascule RS avec portes NOR

La table de vérité est très voisine de celle que l'on a obtenue avec les portes NAND.

| $R$ | $S$ | $Q_{n(t)}$ | $\overline{Q_{n(t)}}$ |
|---|---|---|---|
| 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 0 | 0 | $Q_{n-1(t)}$ | $\overline{Q_{n-1(t)}}$ |
| 1 | 1 | indéterminé | indéterminé |

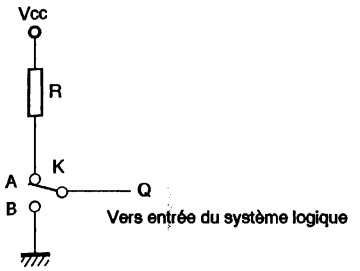
L'état des sorties est mémorisé lorsque $R = S = 0$, et n'est pas prévisible lorsque $R = S = 1$. Cette configuration doit être interdite.

### Application simple

Dans les systèmes numériques, il faut souvent faire l'acquisition d'informations. Le cas le plus simple consiste en une touche d'un clavier mais il peut aussi s'agir d'un contacteur de fin de course dans un asservissement par exemple.

Un contacteur est utilisé pour transformer l'information mécanique en information électrique : c'est le changement d'état logique d'une ligne d'entrée.

La figure 2.2 illustre ce procédé. Au repos, l'interrupteur $K$ est dans la position $A$. La sortie $Q$, qui est l'entrée logique du système, est à l'état haut.



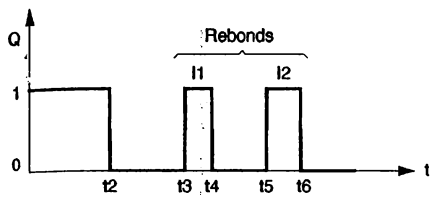




Fig. 2.2
Rebonds occasionnés par un contact.

Lorsque *K* change de position et passe de l'état *A* vers l'état *B*, la forme de la tension présente à la sortie *Q* est représentée à la figure 2.2.

La sortie *Q* passe à 0 à l'instant $t_2$, puis entre les instants $t_3$ et $t_4$, puis $t_5$ et $t_6$ et repasse fugitivement à l'état haut.

Les impulsions correspondantes, $I_1$ et $I_2$, sont dites **rebonds**.

Ces rebonds sont dûs uniquement aux imperfections mécaniques du commutateur *K*. Si le commutateur est parfait, ce qui n'est jamais le cas, les rebonds n'existent pas.

À la figure 2.2, seuls deux rebonds sont représentés. Dans la pratique ceux-ci peuvent être beaucoup plus nombreux et d'amplitude variable.

Après le temps $t_6$, le commutateur *K* est stabilisé dans la position *B*.

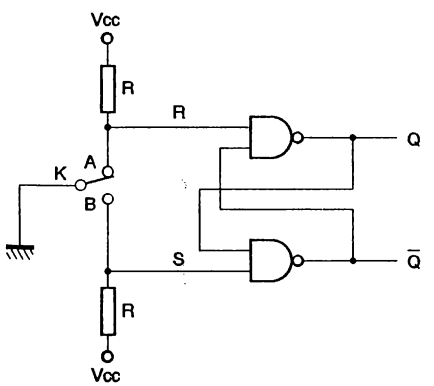
La configuration de la figure 2.2 est inutilisable par un système logique. En effet, si l'on suppose que *K* est la touche d'un clavier, le signal obtenu correspondrait à trois impulsions successives sur cette touche : de $t_2$ à $t_3$ puis de $t_4$ à $t_5$ et pour les temps postérieurs à $t_6$.

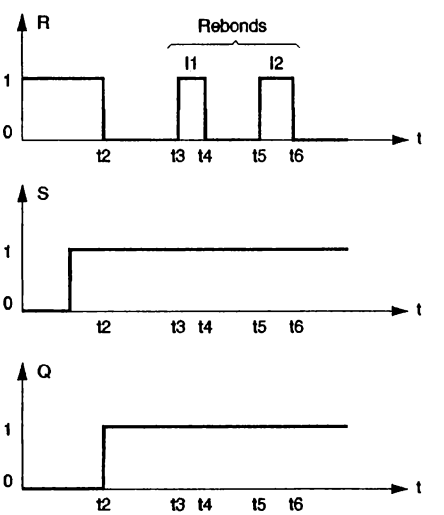
Un système connecté à la sortie *Q* pourrait interpréter l'information comme une suite de trois passages au zéro bien qu'il n'y ait, en fait, qu'un seul et unique passage.

Ceci peut entraîner de graves erreurs si ce signal est, par exemple, envoyé directement à l'entrée interruption d'un microcontrôleur.

Une bascule RS connectée conformément au schéma de la figure 2.3 permet l'élimination des rebonds.

Fig. 2.3
Élimination des rebonds occasionnés par un contact.



En position de repos, l'interrupteur $K$ est sur la position $A$, avec $R = 1$ et $S = 1$.

La sortie $Q$ est au niveau bas : $Q = 0$.

Lorsque l'interrupteur $K$ passe de la position $A$ vers la position $B$, le signal $R$ est entaché des mêmes rebonds que précédemment. À partir de l'instant $t_2$, la bascule change d'état et la sortie $Q$ passe au 1 logique. Les rebonds, temps entre $t_3$ et $t_4$ puis entre $t_5$ et $t_6$, n'ont plus d'effet sur l'état de la bascule puisqu'ils correspondent aux conditions d'entrée : $R = S = 1$.

Ces conditions n'affectent pas l'état des sorties.

## 3 Bascule RS synchronisée

La bascule RS simple est rarement utilisée dans les circuits logiques. L'application la plus courante est celle qui a été vue précédemment : l'interface avec un contacteur.

Dans la pratique, on utilise d'autres configurations construites autour des bascules RS. La première de ces configurations est la bascule RS synchronisée représentée à la figure 2.4.

Il est facile de vérifier que la table de vérité est la suivante :

| $S$ | $R$ | Clk | $Q_{n(t)}$ | $\overline{Q_{n(t)}}$ |
|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 |
| 0 | 0 | 1 | $Q_{n-1(t)}$ | $\overline{Q_{n-1(t)}}$ |
| 1 | 1 | 1 | indéterminé | indéterminé |
| * | * | 0 | $Q_{n-1(t)}$ | $\overline{Q_{n-1(t)}}$ |

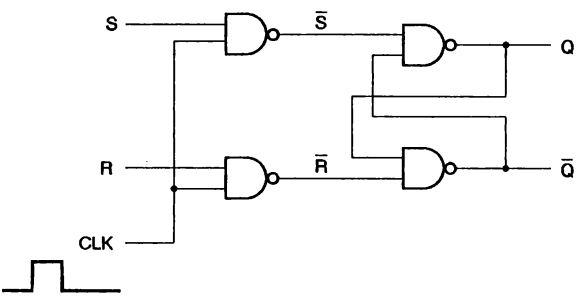



Fig. 2.4
Bascule RS synchronisée.

Lorsque le signal d'horloge CLK est à l'état haut, les données présentes aux entrées $R$ et $S$ sont inversées et transmises aux sorties $Q$ et $\overline{Q}$. Lorsque le signal d'horloge CLK passe à l'état bas, les deux entrées $\overline{S}$ et $\overline{R}$ de la bascule passent à l'état haut et les données antérieures sont mémorisées.

## Bascule D

| D | G | $Q_{n(t)}$ | $\overline{Q_{n(t)}}$ |
|---|---|---|---|
| 0 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 |
| * | 0 | $Q_{n(t)}$ | $\overline{Q_{n-1(t)}}$ |

Cette première variante constitue une étape vers la bascule de la figure 15.5 qui est une véritable cellule **mémoire un bit**.

Pour cette bascule, la table de vérité est donnée ci-contre.

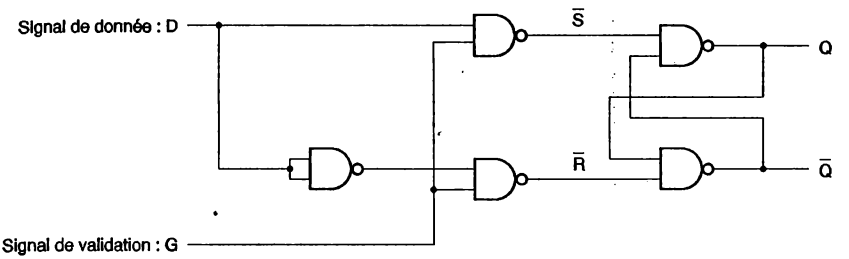



Fig. 2.5

Cellule mémoire un bit transférant les données en sortie si le signal de validation est à 1.

Tant que l'entrée de validation *G* est au niveau haut, les données présentes à l'entrée *D* sont transférées à la sortie *Q*. Dès que l'entrée de validation *G* passe à l'état bas, les données d'entrée ne sont plus prises en compte et les sorties conservent l'état précédant le passage de *G* au zéro.

Pour les bascules représentées en 2.4 et 2.5, il faut faire une remarque importante. Les sorties *Q* et $\overline{Q}$ peuvent changer d'état en même temps que les entrées tant que le signal de validation, CLK ou *G*, est à l'état haut.

Cette caractéristique vaut à ces bascules le qualificatif de « transparentes », parfois utilisé.

Dans la plupart des cas, on préfère des bascules pour lesquelles les données sont transférées au moment du changement d'état du signal de validation, front montant ou front descendant du signal d'horloge.

### Bascule maître-esclave déclenchée par les fronts

Ce type de bascule est le plus couramment utilisé. L'état des sorties *Q* et $\overline{Q}$ est fonction de l'état d'entrée juste avant une transition du signal horloge.

La structure interne d'une **bascule *D* maître-esclave** est donnée à la figure 2.6. À partir de ce schéma, on peut examiner le fonctionnement de la bascule.

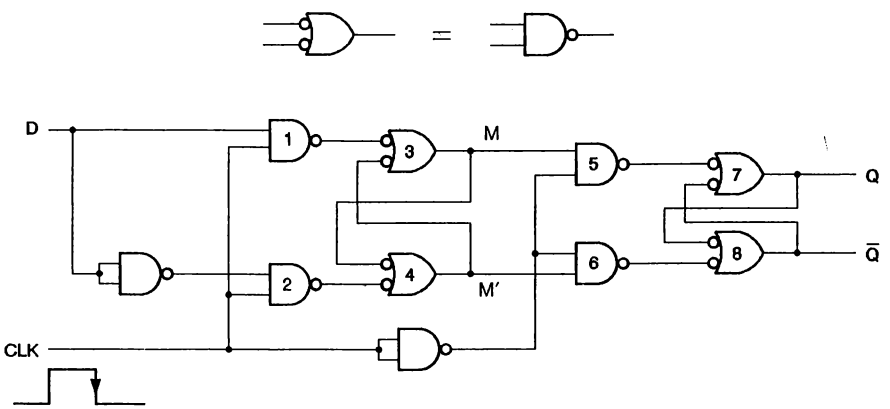



Fig. 2.6
Bascule *D* maître-esclave.

Lorsque le signal d'horloge est à l'état haut, les portes 1 et 2 sont validées. Les données d'entrée *D* sont alors transférées à la sortie de la bascule maître constituée des portes 3 et 4.

Le niveau présent à la sortie *M* est donc l'image du niveau présent à l'entrée *D*, et change en même temps que *D*.

Les portes 5 et 6 sont inhibées, les niveaux *M* et $\overline{M}$ n'ont donc aucune influence sur l'état de sorties *Q* et $\overline{Q}$. La bascule esclave constituée des portes 7 et 8 a donc conservé l'état précédent.

Lorsque le signal d'horloge passe à l'état bas, l'information présente à l'entrée *D* est mémorisée par la bascule maître et transmise à la bascule esclave.

La sortie *Q* est donc l'image de l'entrée *D* juste avant la transition haut-bas du signal d'horloge.

Tant que le signal horloge reste à l'état bas, la donnée présente au point *M* n'évolue pas. Si le signal d'horloge repasse à l'état haut, la donnée, présente au point *M*, suit la donnée *D* mais la bascule esclave est inhibée.

Ceci montre que seul le front de descente du signal CLK a eu une influence sur l'état de la bascule.

Ce type de bascule est appelé bascule *D*.

La figure 2.7 donne un autre exemple de réalisation d'une bascule *D*. Dans ce cas il n'y a pas de notion de maître-esclave mais simplement trois bascules RS combinées.

Supposons qu'à l'origine, la donnée soit zéro et que le signal CLK soit au niveau bas. Les entrées de la bascule RS principale – portes 5 et 6 – sont au niveau 1, l'état antérieur est donc conservé.

Si le signal CLK passe à l'état 1, la sortie de la porte 2 passe au niveau zéro et cette donnée est transférée à la sortie Q. Si le signal horloge repasse au niveau bas, les entrées de la bascule RS principale repassent au niveau haut, n'ayant aucune influence sur l'état de la bascule.

Le niveau zéro – niveau bas – sur l'entrée *D* a été mémorisé à l'instant du front montant du signal d'horloge CLK.

Si maintenant la donnée vaut 1, comme précédemment les deux entrées de la bascule RS principale sont au niveau haut n'ayant aucune influence sur *Q* et $\overline{Q}$. Si le signal d'horloge CLK passe à l'état haut, la sortie de la porte 3 passe au niveau bas et ce niveau est transféré sur la sortie $\overline{Q}$.

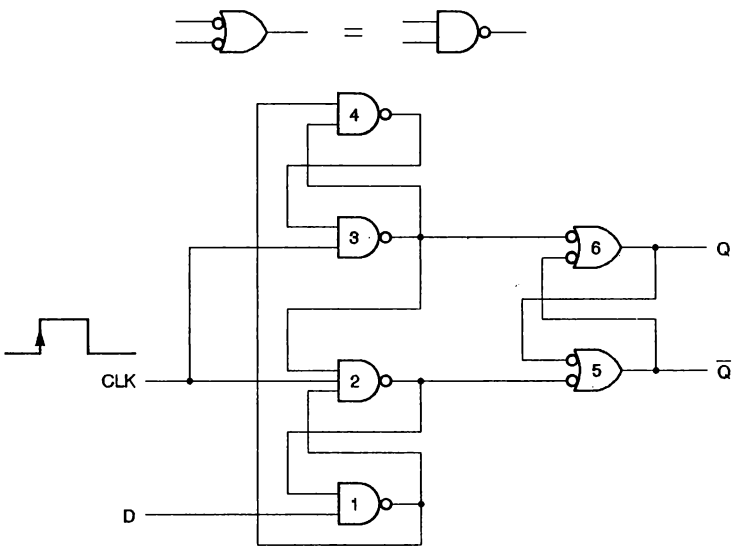



Fig. 2.7
Bascule *D* maître-esclave.

Comme précédemment, la donnée a été mémorisée par le front montant du signal d'horloge.

Si le signal d'horloge repasse à l'état bas, la sortie de la porte 3 repasse au niveau haut, ce qui n'affecte pas l'état des sorties et montre que seul le front montant a un effet.

## 2 Entrées de prépositionnement et remise à zéro

Dans les systèmes logiques, il est souvent utile de pouvoir forcer l'état d'une bascule et ceci indépendamment d'un signal d'horloge.

À la mise sous tension, on veut par exemple s'assurer qu'une ou plusieurs bascules *D* soient au niveau bas ou au niveau haut. Pour obtenir cette fonction, on dispose deux entrées auxiliaires (figure 2.8) qui permettent une action directe sur la bascule esclave.

L'entrée de **remise à zéro** est appelée RAZ, RST ou RESET et l'entrée de **prépositionnement** à 1, PRESET ou SET.

Au schéma de la figure 2.8, en mode normal, ces entrées sont inactives au niveau haut et actives à l'état bas.

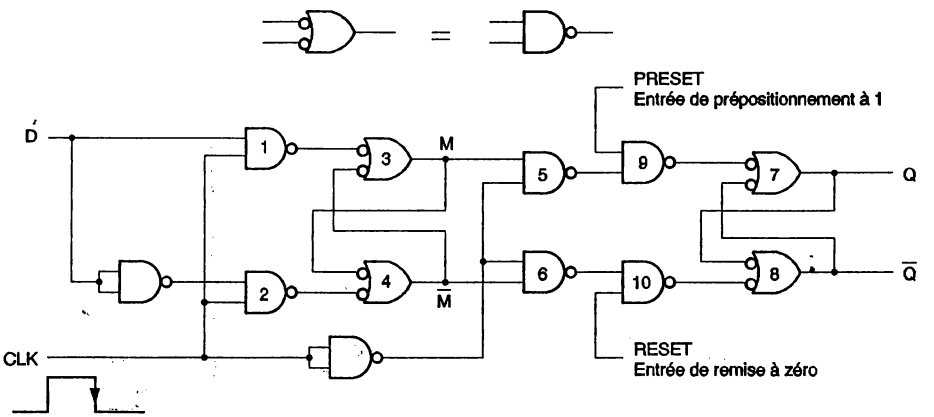



Fig. 2.8
Bascule D maître-esclave avec RESET et PRESET.

Ceci signifie qu'un niveau bas sur l'entrée RESET introduit un zéro dans la bascule RS esclave et qu'un niveau bas sur l'entrée SET introduit un niveau haut.

La remise à zéro ou la mise à 1 de la bascule esclave sont indépendantes des signaux d'entrées *D* ou CLK.

Pour cette raison, les entrées SET et RESET sont dites asynchrones par opposition avec l'introduction des données par l'entrée *D* qui est effectuée en synchronisation avec le signal d'horloge CLK.

## Bascule JK maître-esclave

La bascule *JK* maître-esclave est très voisine de la bascule *D* comme le montre la figure 2.9. À la différence de la bascule *D*, la bascule *JK* possède deux entrées. Pour cette bascule, la table de vérité est la suivante :

| J | K | $Q_{n(t)}$ |
|---|---|---|
| 0 | 0 | $Q_{n-1(t)}$ |
| 0 | 1 | 0 |
| 1 | 0 | 1 |
| 1 | 1 | Changement d'état à chaque coup d'horloge. |

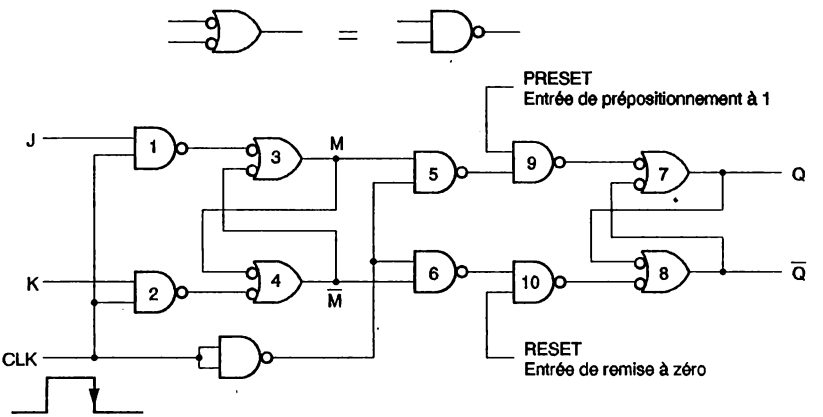



Fig. 2.9
Bascule JK maître-esclave avec RESET et PRESET.

Si *J* et *K* sont complémentaires, la sortie *Q* prend la valeur de *J*. Si *J* et *K* sont à l'état bas, la sortie ne change pas d'état et si *J* et *K* sont à l'état haut, la sortie change d'état à chaque coup d'horloge.

Comme pour la bascule *D*, des entrées Set et Reset peuvent être ménagées sur la bascule *JK*.

Les figures 2.10 et 2.11 représentent la structure interne de deux circuits intégrés populaires de la famille TTL 74 112 : double bascule *JK*, et 7474 : double bascule *D*. Chaque structure est accompagnée de la table de vérité représentative du fonctionnement du circuit.

Ces deux circuits sont donnés à titre d'exemple. Il existe d'autres types de bascule dans les familles CMOS, HCMOS, ECL, etc.

FUNCTION TABLE

| INPUTS | | | | | OUTPUTS | |
|---|---|---|---|---|---|---|
| Preset | Clear | Clock | J | K | Q | $\overline{Q}$ |
| L | H | X | X | X | H | L |
| H | L | X | X | X | L | H |
| L | L | X | X | X | H* | H* |
| H | H | ↓ | L | L | $Q_0$ | $\overline{Q}_0$ |
| H | H | ↓ | H | L | H | L |
| H | H | ↓ | L | H | L | H |
| H | H | ↓ | H | H | TOGGLE | |
| H | H | H | X | X | $Q_0$ | $\overline{Q}_0$ |

DUAL J-K NEGATIVE-EDGE-TRIGGERED FLIP-FLOPS WITH PRESET AND CLEAR

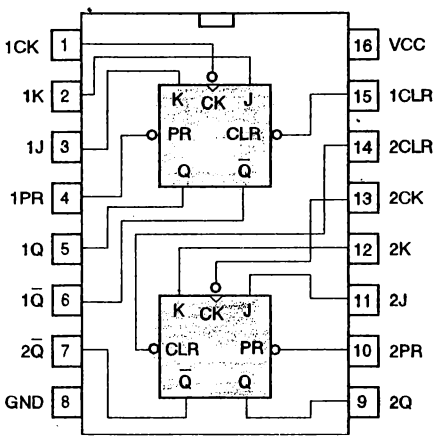



Fig. 2.10
Double bascule *JK* 74112.

FUNCTION TABLE

| INPUTS | | | | OUTPUTS | |
|---|---|---|---|---|---|
| Preset | Clear | Clock | D | Q | $\overline{Q}$ |
| L | H | X | X | H | L |
| H | L | X | X | L | H |
| L | L | X | X | H* | H* |
| H | H | ↑ | H | H | L |
| H | H | ↑ | L | L | H |
| H | H | L | X | $Q_0$ | $\overline{Q}_0$ |

DUAL D-TYPE POSITIVE-EDGE-TRIGGERED FLIP-FLOPS WITH PRESET AND CLEAR

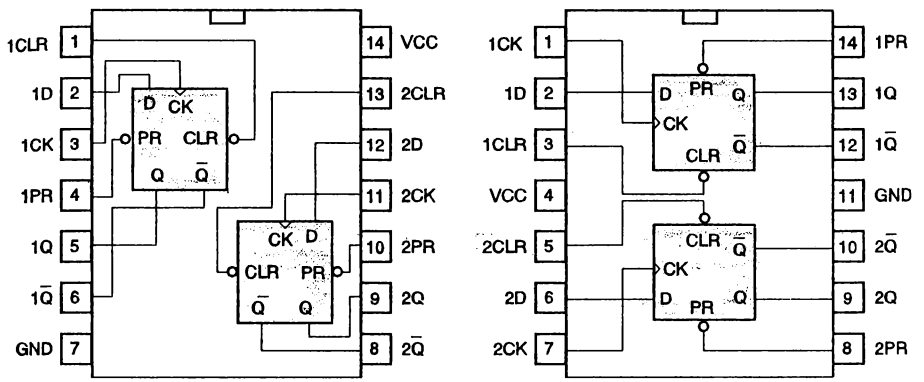



Fig. 2.11
Double bascule *D* 7474.

## 1 Limitation en fréquence et temps de propagation

Le diagramme de la figure 2.12 met en évidence les paramètres dynamiques de la bascule *JK* prise en exemple, 74 LS 112.

Cette bascule fonctionne de manière garantie avec une fréquence d'horloge comprise entre 0 et 30 MHz, $f_c \leq 30$ MHz.

La largeur de l'impulsion d'horloge minimale ainsi que les largeurs minimales des impulsions Clear et Preset sont respectivement de 20 et 25 ns.

Les données appliquées aux entrées *JK* doivent être stables pendant un temps $t_{su}$ avant le front de descente de l'horloge. Ce temps $t_{su}$, dit temps de préaffichage, vaut 20 ns au minimum.

Enfin, on indique le temps de propagation des différentes entrées vers les sorties $Q$ et $\overline{Q}$.

Ces temps sont spécifiés en valeur typique et valeur maximum, typique 15 ns et maximum 20 ns.

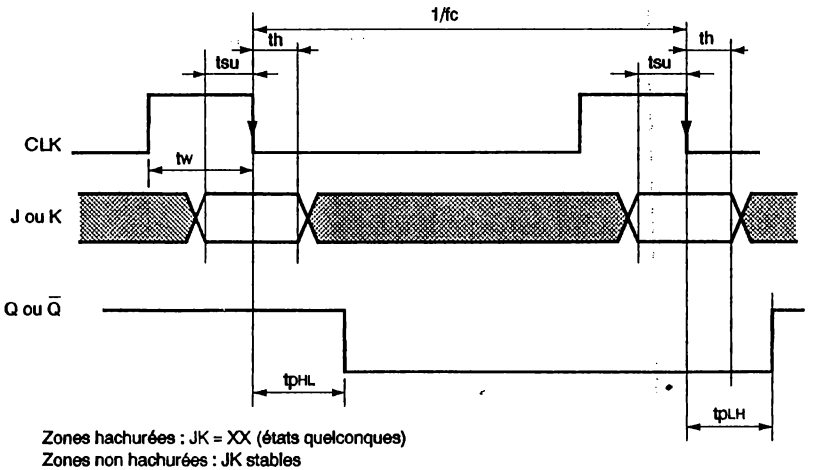



Fig. 2.12
Paramètres dynamiques de la bascule *JK*.

Pour certains circuits, on spécifie un temps de maintien $t_h$ qui indique que les informations d'entrée doivent rester affichées un certain temps après le front actif de l'horloge.

Pour la bascule 74LS 112 prise en exemple, ce temps vaut 0 ns. Ce qui signifie que les données d'entrées peuvent changer simultanément avec le front descendant du signal d'horloge.

Les paramètres temporels doivent être pris en compte par le concepteur pendant l'étude de la circuiterie logique. La famille logique employée TTL LS, CMOS, HCMOS, ECL, est sélectionnée en fonction de la fréquence maximale des signaux dans l'application.

---

**Application des bascules**

Les bascules *D* ou bascules *JK*, assemblées de manière particulière, permettent la réalisation de fonctions plus complexes comme les registres à décalage et les compteurs.

La suite de ce chapitre est consacrée à la description de ces deux fonctions et à quelques-unes de leurs applications.

---

## Registres à décalage

Une bascule est une mémoire 1 bit. En groupant *N* bascules du même type, on constitue une mémoire *N* bits. Une telle combinaison est appelée un registre. La sortie de chaque bascule est connectée à l'entrée de la bascule suivante. La figure 2.13 repré-

sente une configuration appelée registre à décalage et son registre permet de stocker 5 bits.

Chacune des bascules élémentaires peut être du type *RS* ou *JK* et la première bascule est connectée en type *D*.

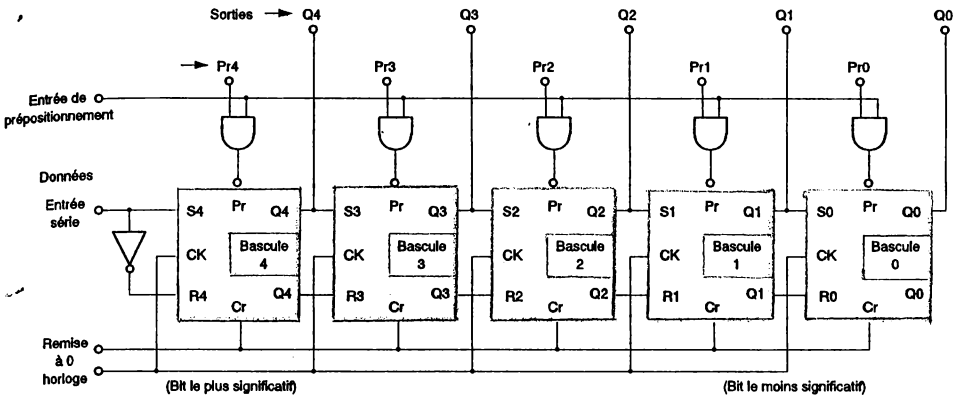



Fig. 2.13
Registre à décalage 5 bits.

## 1 Registre à entrée série et sortie parallèle

Pour expliquer le fonctionnement du registre de la figure 2.13, supposons que l'on désire stocker le mot 01011 dans le registre.

Le bit le moins significatif est situé le plus à droite tandis que le plus significatif est situé à gauche.

Pour stocker ce message, il faut générer les signaux de la figure 2.14 regroupant d'une part le signal de données et d'autre part le signal horloge. On suppose que les bascules sont actives sur le front descendant du signal d'horloge.

Les zones hachurées indiquent que les données n'ont pas d'importance pendant ces instants.

La bascule est préalablement initialisée en appliquant un niveau bas sur l'entrée remise à zéro. Après cette initialisation, les sorties $Q_0$ à $Q_4$ sont au niveau bas.

La première donnée est présentée à l'entrée donnée. Sur le front descendant du signal d'horloge, cette donnée est mémorisée par la bascule 4. Le mot 00001 apparaît alors sur les sorties $Q_0$ à $Q_4$.

La deuxième donnée est introduite et au deuxième coup d'horloge, on obtient 00011.

Au troisième coup d'horloge, 00110.

Au quatrième coup d'horloge, 01101, et finalement au cinquième coup, 11010.

Nous obtenons finalement :

$$Q_0 = 1,\ Q_1 = 1,\ Q_2 = 0,\ Q_3 = 1 \text{ et } Q_4 = 0.$$

À chaque coup d'horloge, l'information présente à la sortie de la bascule $N$ est décalée et stockée dans la bascule $N - 1$. Cette caractéristique de décalage vaut à ce registre le nom de registre à décalage.

La structure de la figure 2.13 a permis de réaliser une conversion série parallèle. Les données ont été introduites en série, les unes derrière les autres, et sont maintenant disponibles simultanément, en parallèle, sur les sorties $Q_0$ à $Q_4$.

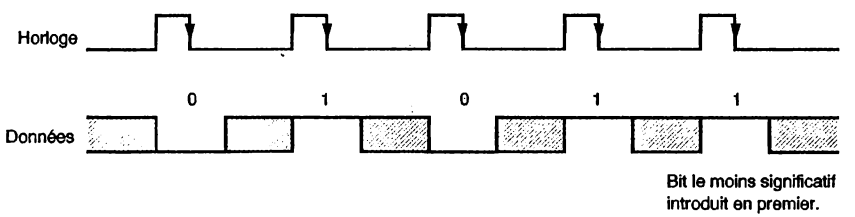



Fig. 2.14
Signaux d'entrée pour mémoriser le mot 01011 dans le registre de la figure 2.13.

## 2 Registre à entrée série et sortie série

Supposons que le mot pris en exemple, 01011, soit stocké dans le registre de la figure 2.13.

Si maintenant cinq nouvelles impulsions d'horloge sont envoyées au circuit, le mot initial peut être récupéré en **série** sur la sortie $Q_0$.

Il faudra $N$ impulsions d'horloge pour un mot de $N$ bits. Après la énième impulsion d'horloge, chaque bascule lit la valeur 0 si l'entrée donnée est au niveau bas.

Les deux cadences ou fréquences de mémorisation et relecture peuvent être différentes. Le registre à entrée série/sortie série permet donc la modification de l'espacement temporel entre les données.

Il s'agit donc d'une modification de la vitesse du débit des informations.

## 3 Registre à entrées parallèles et sortie série

Le registre à décalage permet finalement la conversion d'un mot présenté en parallèle en un message transmis en série.

Supposons que l'on dispose, en sortie d'un système quelconque, d'un mot 01011 en parallèle et que l'on souhaite le transmettre en série. Pour cette conversion, on utilisera le registre de la figure 2.13. Le bit le moins significatif est appliqué à $Pr_0$ et le bit le plus significatif à $Pr_4$.

Toutes les bascules sont préalablement remises à zéro. Ensuite, l'entrée de prépositionnement passe momentanément à l'état bas.

La bascule 0 a mémorisé le bit le moins significatif, la bascule 1 le second bit et ainsi de suite jusqu'au bit 5 mémorisé dans la bascule 4.

Il suffit alors d'appliquer cinq impulsions d'horloge à l'entrée horloge pour disposer du message à la sortie $Q_0$.

## 4 Registre à entrées parallèles et sorties parallèles

Dans le cas précédent, lorsque la bascule est initialisée par remise à zéro, puis prépositionnée, les sorties $Q_0$ à $Q_4$ ont mémorisé les états appliqués sur les entrées $Pr_0$ à $Pr_4$.

Dans ce cas, il n'y a plus de notion de décalage, le dispositif ne fonctionne pas comme un registre à décalage puisqu'aucune impulsion d'horloge n'est nécessaire.

Chaque bascule sert simplement de mémoire vive distincte d'un bit.

## 5 Registre à décalage universel

Le registre 74 194 est un registre dit universel car il autorise toutes les configurations citées en exemple :
- entrées parallèles, sorties parallèles,
- entrée série, sorties parallèles,
- entrées parallèles, sortie série,
- entrée série, sortie série.

En outre, le décalage s'effectue, soit de la gauche vers la droite, soit de la droite vers la gauche. La structure interne de ce registre est représentée à la figure 2.15.

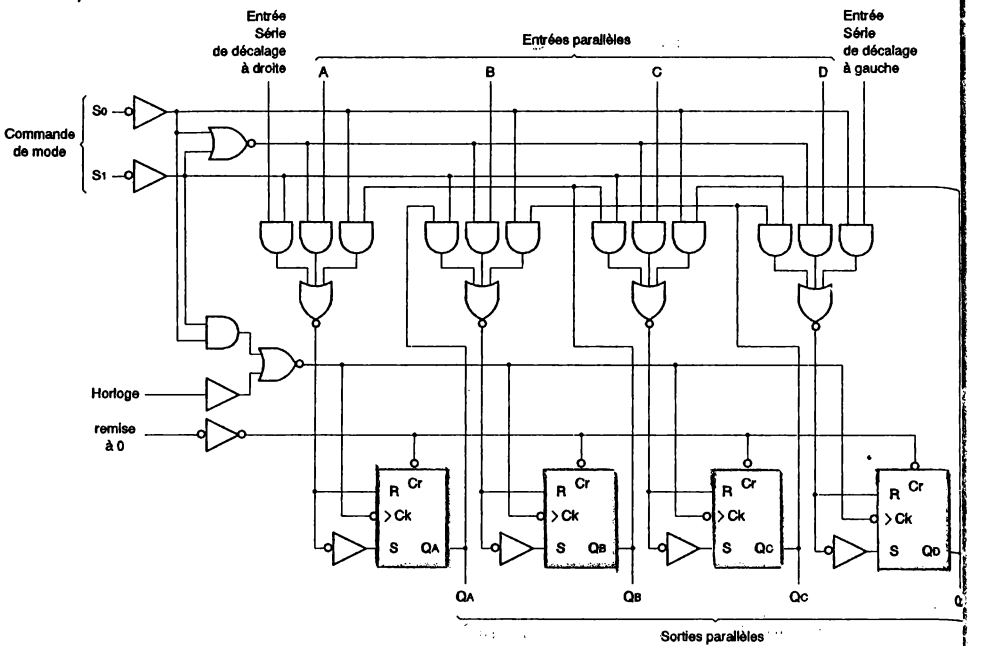



Fig. 2.15
Structure du registre universel 74194.

Deux entrées de mode permettent de définir le fonctionnement :

| $S_0$ | $S_1$ | Mode |
|---|---|---|
| 0 | 0 | invalidation : mémorisation des données |
| 1 | 1 | entrées parallèles |
| 1 | 0 | décalage à droite |
| 0 | 1 | décalage à gauche |

Le circuit intégré 74 194 est un registre 4 bits et plusieurs circuits peuvent être groupés pour des mots de longueur plus importante.

Il existe d'autres applications pour les registres à décalage. Deux applications spécifiques ont été retenues pour montrer l'importance de ces circuits.

# Applications des registres à décalage

## 1 Circuit de retard numérique

Une succession de registres à décalage permet la réalisation simple d'une ligne à **retard numérique** comme le montre la figure 2.16.a.

Le circuit CMOS 4562 est constitué par la mise en série de 128 bascules de type *D* (figure 2.16.b) .

Soit une impulsion d'entrée de largeur $T_{WE}$ présente à l'entrée données et un signal d'horloge de période *t*.

Supposons que l'on observe le signal de sortie, en sortie de la 128ème bascule *D*. L'impulsion d'entrée est retardée d'un temps *T* :

$$T = 128 \cdot t$$

La largeur de l'impulsion de sortie $T_{WS}$ est limitée à la partie entière de la largeur $T_{WE}$ divisée par la période du signal d'horloge :

$$T_{WS} = t \cdot \text{int}\left(\frac{T_{WE}}{t}\right)$$

Le signal d'horloge et les impulsions d'entrée ne sont pas synchronisés. En conséquence le retard *T* n'est pas constant et vaut $T + T_1$ avec $0 \leq T_1 \leq t/2$.

Le retard n'est connu qu'avec une erreur ou précision $t/2$.

Pour minimiser cette erreur, la fréquence horloge doit être la plus grande possible. Pour des retards *T* importants, ceci entraîne un nombre important de bascules. Plusieurs circuits 4562 peuvent alors être mis en série.

Cette circuiterie possède en outre la caractéristique intéressante de synchroniser des impulsions d'entrée et de les calibrer à une largeur multiple de la fréquence horloge.

## 2 Générateur pseudo-aléatoire

Un générateur est en général un outil de référence. Qu'il soit générateur de tension, courant ou fréquence, il est utilisé comme étalon.

Sa première caractéristique est sa stabilité dans le temps. Aux instants *t* et $t + \Delta t$, la grandeur de sortie n'aura pas évolué de plus de $10^{-6}$ ou $10^{-7}$ par exemple.

Cette caractéristique autorise les relevés respectifs et comparatifs des performances d'un matériel.

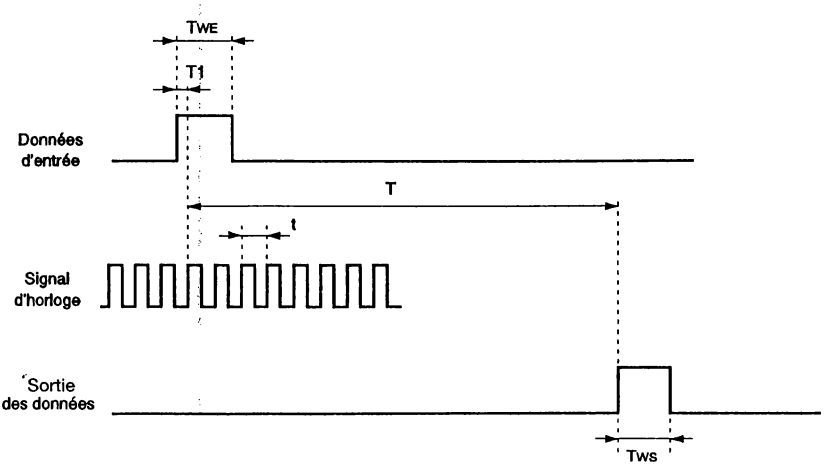



Fig. 2.16.a.
Principe de réalisation du retard numérique.

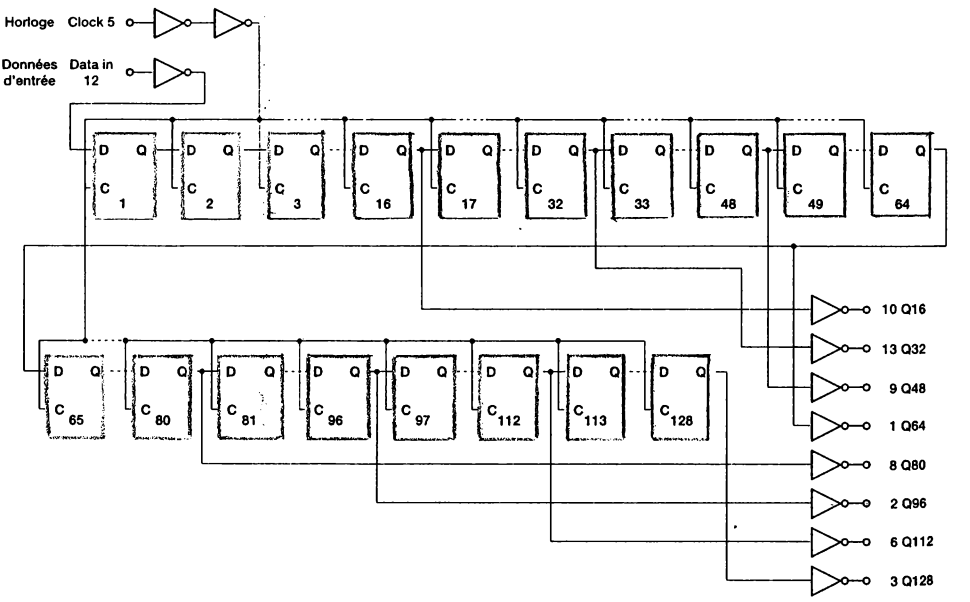



Fig. 2.16.b
Réalisation d'un retard numérique avec un circuit CMOS 4562.

À contrario, un générateur aléatoire délivre une grandeur de sortie qui, à l'instant $t + \Delta t$, ne dépend aucunement de la grandeur de sortie à l'instant $t$. Pour un générateur parfaitement aléatoire, la grandeur de sortie serait une quantité qui pourrait prendre une valeur quelconque d'un ensemble spécifié, ce que l'on peut assimiler à une succession de tirages au hasard. Un générateur de bruit blanc sera classé dans la catégorie des générateurs aléatoires.

Un générateur pseudo-aléatoire,comme son nom l'indique, n'est pas un générateur parfaitement aléatoire, puisque la grandeur de sortie à l'instant $t + \Delta t$ dépend de la grandeur de sortie à l'instant $t$.

Mais si l'on examine les grandeurs de sortie, celles-ci donnent l'**apparence** d'une distribution parfaitement aléatoire.

La figure 2.17 représente un registre à décalage dont la sortie $Q_0$ est rebouclée sur l'entrée.

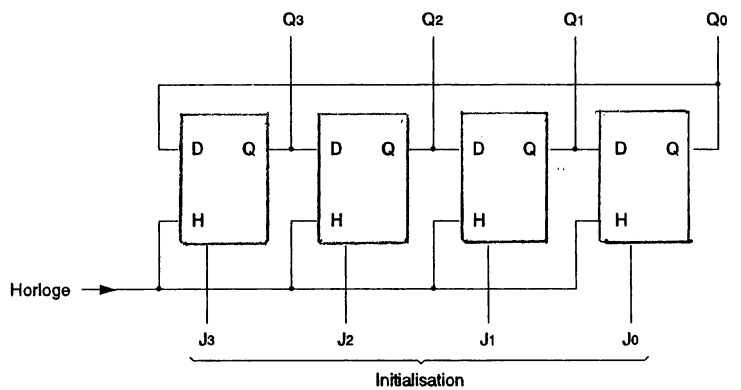



Fig. 2.17
Rebouclage de quatre bascules *D*.

L'initialisation consiste à charger les données $J_0$ à $J_3$ dans les bascules. Après cette phase d'initialisation, le générateur peut entrer dans son mode de fonctionnement normal : l'activation par un signal d'horloge.

Supposons que le registre soit initialisé par le mot 0100. Après la première impulsion d'horloge, le mot de sortie vaut 0010, puis à la seconde impulsion 0001 et à la troisième 1000. On réalise ainsi une permutation circulaire et le régime établi est donc élémentaire.

Si l'on exprime le mot de sortie en hexadécimal, dans l'exemple précédent, le système passe par quatre états différents : 4, 2, 1 et 8 avant de revenir au mot d'initialisation 4.

La séquence est la suivante: 4, 2, 1, 8, 4, 2, 1, 8 et 4, etc.

Ce premier exemple simple montre que le rebouclage ne permet qu'un nombre de combinaisons de sortie égal au nombre de registres à décalage. La période vaut 4 dans ce cas.

La séquence obtenue en sortie dépend du mot d'initialisation.

La figure 2.18 met en œuvre un rebouclage différent, effectué par un OU exclusif. Comme précédemment, $J_0$ et $Q_0$ sont les bits les moins significatifs et les mots sont exprimés en hexadécimal.

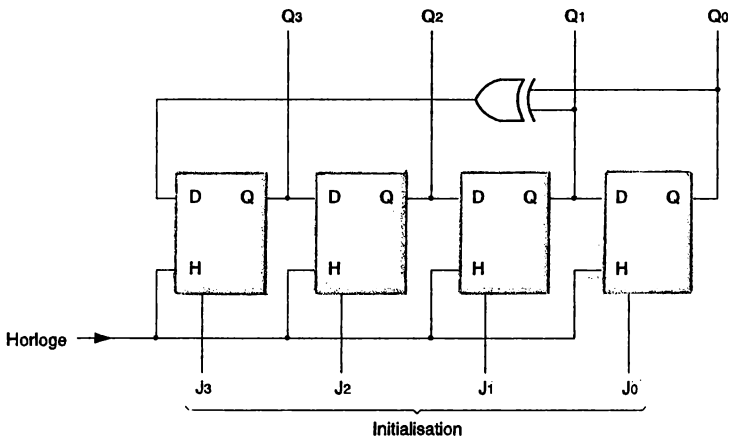



Fig. 2.18
Rebouclage du registre à décalage avec une porte OU exclusif.

Supposons que le système soit initialisé par la valeur 5. Pour le mot de sortie, on obtient la séquence suivante : 5, A, D, E, F, 7, 3, 1, 8, 4, 2, 9, C, 6, B, 5 et A,

Si le système est initialisé par 9, l'ordre de la séquence reste inchangé, la séquence est identique à la séquence précédente, seul le point de départ évolue.

Dans le cas de la figure 2.18, on remarque que la période de la séquence vaut 15.

On peut remarquer que la succession des grandeurs de sortie **ressemble** à une distribution aléatoire : succession de tirages au hasard.

Si les entrées de la porte OU exclusif sont connectées aux sorties $Q_0$ et $Q_2$ des registres, la période est fonction du mot d'initialisation et vaut soit 3 soit 6.

Pour le registre à décalage rebouclé sur lui-même, le nombre d'états de sortie est égal au nombre de bascules, si le rebouclage est effectué par une porte OU exclusif, la périodicité est variable et fonction du mot d'initialisation.

On remarque que le mot d'initialisation correspondant à zéro doit être éliminé puisqu'il ne se passe rien (état permanent sur les sorties).

Pour le générateur pseudo-aléatoire, on cherche le rebouclage qui donnera la **périodicité maximale**.

### *2.1 Équations du système*

Le mot de sortie à l'instant $t + \Delta t$ dépend du mot de sortie à l'instant $t$. Il est donc commode de définir une opération qui relie deux mots consécutifs.

$Q_{n(t)}$ et $Q^{*}_{n-1(t)}$ sont deux vecteurs qui comportent autant de composantes qu'il existe de registres à décalage dans la structure.

Une matrice de passage $A$ permet le calcul de $Q_{n-1(t)}$ :

$$Q_{n+1(t)} = A \cdot Q_{n(t)}$$

La matrice $A$ est définie de la manière suivante :

$$\begin{bmatrix} a_1 & a_2 & a_3 & a_{K-1} & a_K \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ . & . & . & . & . \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Les coefficients $a_1, a_2, a_3, \ldots, a_K$ sont choisis égaux à 0 ou 1 et les opérations sont effectuées dans l'arithmétique modulo 2 (résultats égaux à 0 ou 1).

Les résultats suivants sont donnés sans la démonstration qui sortirait du cadre de cet ouvrage.

La périodicité du système est liée au polynôme caractéristique de la matrice $A$.

On cherche ce polynôme en calculant le déterminant de la matrice $B$ :

$$B = A - x \cdot I$$

où $I$ représente la matrice unité.

$$P(x) = \det[A - x \cdot I]$$

La matrice *B* est définie de la manière suivante :

$$\begin{bmatrix} a_1 - x & a_2 & a_3 & & a_{K-1} & a_K \\ 1 & -x & 0 & & 0 & 0 \\ 0 & 1 & -x & & 0 & 0 \\ 0 & 0 & 1 & & 0 & 0 \\ . & . & . & & . & . \\ 0 & 0 & 0 & & -x & 0 \\ 0 & 0 & 0 & & 0 & -x \end{bmatrix}$$

Sachant que l'on travaille modulo 2, [− 1 = 1 ], le polynôme $P(x)$ s'écrit finalement :

$$P(x) = x^K + a_1 \cdot x^{K-1} + a_2 \cdot x^{K-2} + \ldots\ldots + a_K$$

On démontre que lorsque le polynôme est choisi parmi les **polynômes primitifs**, la périodicité du système est maximale et cette période vaut :

$$T = 2^K - 1$$

Le tableau de la figure 2.19 représente une liste de polynômes primitifs d'ordre 2 à 22 facilement utilisable pour la concrétisation de générateurs pseudo-aléatoires. Dans ce cas, une seule porte OU exclusif, ayant deux entrées, suffit pour le rebouclage.

| ORDRE n | PERIODE $2^n - 1$ | P (x) |
|---|---|---|
| 2 | 3 | $x^2 + x + 1$ |
| 3 | 7 | $x^3 + x + 1$ |
| 4 | 15 | $x^4 + x + 1$ |
| 5 | 31 | $x^5 + x^2 + 1$ |
| 6 | 63 | $x^6 + x + 1$ |
| 7 | 127 | $x^7 + x^3 + 1$ |
| 9 | 511 | $x^9 + x^4 + 1$ |
| 10 | 1023 | $x^{10} + x^3 + 1$ |
| 11 | 2047 | $x^{11} + x^2 + 1$ |
| 15 | 32767 | $x^{15} + x + 1$ |
| 22 | 4194303 | $x^{22} + x + 1$ |

**Liste complémentaire des polynômes irréductibles primitifs max. 3 termes**

$x^{15} + x + 1$ $x^{23} + x^5 + 1$
$x^{17} + x^3 + 1$ $x^{25} + x^3 + 1$
$x^{18} + x^7 + 1$ $x^{28} + x^3 + 1$
$x^{20} + x^3 + 1$ $x^{29} + x^2 + 1$
$x^{21} + x^2 + 1$ $x^{31} + x^5 + 1$
$x^{22} + x + 1$ $x^{33} + x^{13} + 1$

**Max. 5 termes pour des ordres multiples de 8**

$x^8 + x^4 + x^3 + x^2 + 1$
$x^{16} + x^{12} + x^3 + x + 1$
$x^{24} + x^7 + x^2 + x + 1$

Fig. 2.19
Polynômes premiers pour des ordres inférieurs ou égaux à 24.

Le tableau de la figure 2.20 représente, quant à lui, une liste de polynômes primitifs d'ordre 23 à 33.

| ORDRE n | PERIODE $2^n - 1$ | P (x) |
|---|---|---|
| 23 | 8388607 | $x^{23} + x^5 + 1$ |
| 25 | 33554431 | $x^{25} + x^3 + 1$ |
| 28 | 268435455 | $x^{28} + x^3 + 1$ |
| 29 | 536870911 | $x^{29} + x^2 + 1$ |
| 31 | 2147483647 | $x^{31} + x^5 + 1$ |
| 33 | 8589934591 | $x^{33} + x^{13} + 1$ |

Fig. 2.20
Polynômes primitifs pour les ordres 23 à 33.

## 2.2 *Réalisation pratique de générateurs pseudo-aléatoires*

Supposons que l'on veuille réaliser un générateur pseudo-aléatoire d'ordre 15 en employant le polynôme :

$$P(x) = x^{15} + x + 1$$

Dès que le choix du polynôme est effectué, la matrice de transfert, ainsi que le nombre de registres, sont parfaitement définis.

Dans ce cas : $a_K = a_{K-1} = 1$

Tous les autres coefficients sont nuls.

On dispose autant de registres à décalage que l'ordre du polynôme. Si les données circulent de gauche à droite, le registre situé le plus à gauche représente $x_{K-1}$, le suivant $x_{K-2}$, etc., jusqu'au dernier qui représente le terme constant 1.

Les deux entrées de la porte OU exclusif sont connectées au registre représentant $x$ et 1 comme le montre le schéma de la figure 2.21.

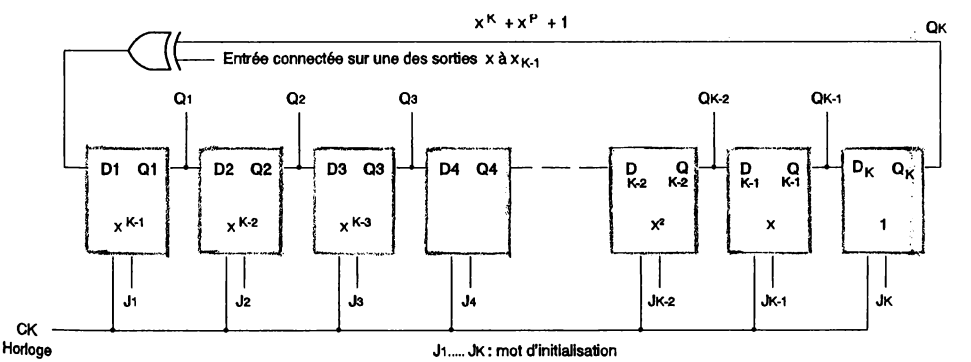



Fig. 2.21
Schéma théorique de base pour réaliser un générateur pseudo-aléatoire utilisant un polynôme.

Tous les états de sortie peuvent être calculés grâce à la relation :

$$\begin{bmatrix} Q_{1\,(n+1)} \\ Q_{2\,(n+1)} \\ \cdot \\ \cdot \\ \cdot \\ Q_{K\,(n+1)} \end{bmatrix} = A \begin{bmatrix} Q_{1\,(n)} \\ Q_{2\,(n)} \\ \cdot \\ \cdot \\ \cdot \\ Q_{K\,(n)} \end{bmatrix}$$

S'il est possible de définir la structure d'un générateur à partir d'un polynôme, l'inverse est également possible.

Avec la structure de la figure 2.18 comportant quatre registres, le polynôme caractéristique vaut :

$$P(x) = x^4 + x + 1$$

En se reportant au tableau de la figure 2.19, on constate que $P(x)$ est bien un polynôme primitif.

Si la porte ou exclusif est connectée à $Q_0$ et $Q_2$, le polynôme caractéristique est :

$$P'(x) = x^4 + x^2 + 1$$

Ce polynôme n'est pas un polynôme primitif et dans ces conditions, le générateur n'a pas la périodicité maximale.

## 2.3 *Utilisation des générateurs pseudo-aléatoires GPA*

Les GPA peuvent être utilisés en transmission, par exemple en **étalement de spectre**, lorsque les symboles 1 ou zéro à transmettre sont remplacés par une séquence pseudo-aléatoire.

Le message de sortie peut être utilisé pour tester une liaison numérique et mesurer le taux d'erreur bit.

Le schéma de la figure 2.22 donne l'exemple d'un GPA connecté à un filtre et un amplificateur, constituant ainsi un **générateur de bruit analogique** basse fréquence.

Les générateurs pseudo-aléatoires sont aussi utilisés dans les systèmes de cryptage-décryptage, et il s'agit de leur application la plus répandue. Le mot d'initialisation constitue alors la clé.

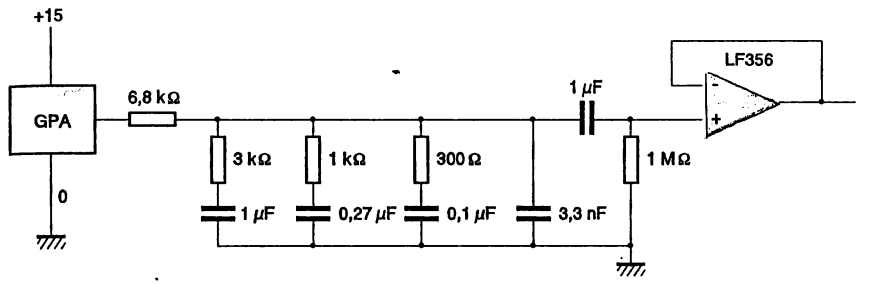



Fig. 2.22
Générateur de bruit blanc à partir d'un générateur pseudo-aléatoire.

## Les compteurs-décompteurs

Un compteur est un ensemble de circuits numériques associant un réseau combinatoire à un ensemble de bascules dont le contenu passe de la valeur $n$ à $n + 1$ après application d'une impulsion d'horloge. Il existe deux types de structures de compteurs : **synchrones** et **asynchrones**.

### 1 Compteurs synchrones

Dans le compteur synchrone représenté à la figure 2.23, l'impulsion d'horloge est appliquée directement à toutes les entrées horloge des bascules.

Un réseau combinatoire calcule les fonctions agissant sur les entrées synchrones associées : *JK* ou *D*.

Dans cette structure, toutes les bascules commutent en même temps après un front déterminé du signal d'horloge.

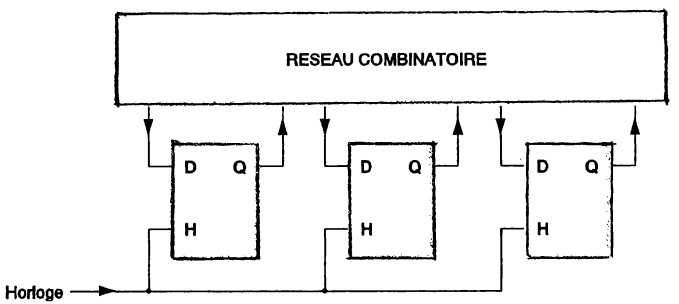



Fig. 2.23
Compteur synchrone.

## 2 Compteurs asynchrones

Dans le compteur asynchrone représenté à la figure 2.24, l'impulsion d'horloge est envoyée uniquement au premier étage. Les autres entrées horloge reçoivent des signaux provenant de l'état des étages du compteur.

De cette façon, les étages ne basculent plus en synchronisme avec l'impulsion d'horloge mais après les transitions des étages qui les pilotent. On dit que la progression est asynchrone.

La synthèse d'un compteur consiste à déterminer les fonctions de commande des bascules dans les structures des figures 2.23 et 2.24.

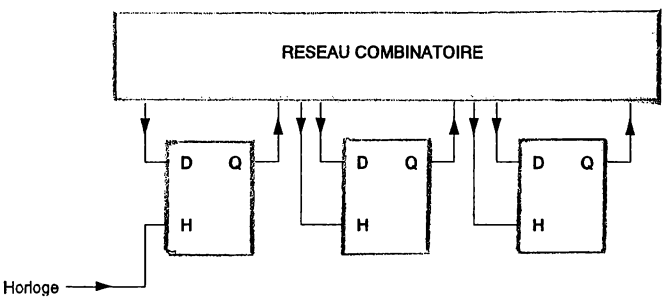



Fig. 2.24
Compteur asynchrone.

## 3 Synthèse des compteurs synchrones

Les exemples qui suivent donnent la démarche à suivre pour effectuer cette synthèse. On souhaite, premièrement, réaliser un circuit diviseur par 3 à l'aide de deux bascules *D*.

Le signal d'horloge est appliqué simultanément aux deux entrées horloge de chaque bascule. Les entrées $D_1$ et $D_2$ sont les entrées des bascules, et $Q_1$ et $Q_2$ leurs sorties.

On choisit premièrement les trois états du compteur :

| Etat | $Q_1$ | $Q_2$ |
|---|---|---|
| 1 | 0 | 0 |
| 2 | 0 | 1 |
| 3 | 1 | 0 |
| 4 | 0 | 0 |

$Q_1 = Q_2 = 0$ à l'état initial, et au quatrième coup d'horloge, le compteur revient dans l'état initial.

À chaque coup d'horloge, les données présentées aux entrées $D$ sont mémorisées par les bascules. En conséquence pour obtenir $Q_1$ et $Q_2$ à l'instant $n$, il faut présenter $Q_1$ et $Q_2$ aux entrées $D$ à l'instant $n - 1$.

Ceci permet l'écriture de la table de vérité suivante :

| $Q_1$ | $Q_2$ | $D_1$ | $D_2$ |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 |

En règle générale, à partir de cette table de vérité et en utilisant la méthode de **Karnaugh**, on en déduit la logique combinatoire.

Dans ce cas simple, on peut remarquer que :

$$D_1 = Q_2$$

et

$$D_2 = \overline{Q_1 + Q_2}$$

Le schéma électrique du compteur diviseur par 3 est représenté à la figure 2.25.

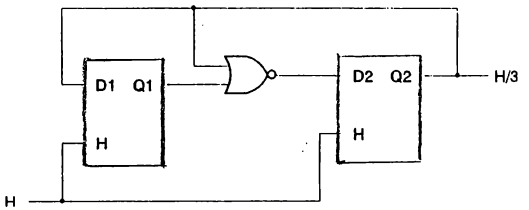



Fig. 2.25
Circuit compteur-diviseur par 3.

Dans le deuxième exemple, l'objectif est de réaliser un compteur binaire pur synchrone à quatre étages.

La table de vérité ci-après donne les différents états du compteur. Ce compteur devra être réalisé avec des bascules *JK*.

L'analyse de cette table de vérité est assez simple. En effet, on constate que $Q_0$ commute systématiquement à chaque coup d'horloge.

La sortie $Q_1$ commute si $Q_0$ est à l'état haut. La sortie $Q_2$ commute si $Q_0$ et $Q_1$ sont à l'état haut. Et finalement, la sortie $Q_n$ commute si tous les étages précédents $Q_{n-1}$ à $Q_0$ sont simultanément à l'état haut.

On peut donc calculer de proche en proche une retenue *R* qui sera utilisée par l'étage suivant.

| Valeur | $Q_3$ | $Q_2$ | $Q_1$ | $Q_0$ |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 2 | 0 | 0 | 1 | 0 |
| 3 | 0 | 0 | 1 | 1 |
| 4 | 0 | 1 | 0 | 0 |
| 5 | 0 | 1 | 0 | 1 |
| 6 | 0 | 1 | 1 | 0 |
| 7 | 0 | 1 | 1 | 1 |
| 8 | 1 | 0 | 0 | 0 |
| 9 | 1 | 0 | 0 | 1 |
| 10 | 1 | 0 | 1 | 0 |
| 11 | 1 | 0 | 1 | 1 |
| 12 | 1 | 1 | 0 | 0 |
| 13 | 1 | 1 | 0 | 1 |
| 14 | 1 | 1 | 1 | 0 |
| 15 | 1 | 1 | 1 | 1 |

Le schéma du compteur binaire synchrone est représenté à la figure 2.26 et il est réalisé avec quatre bascules *JK*.

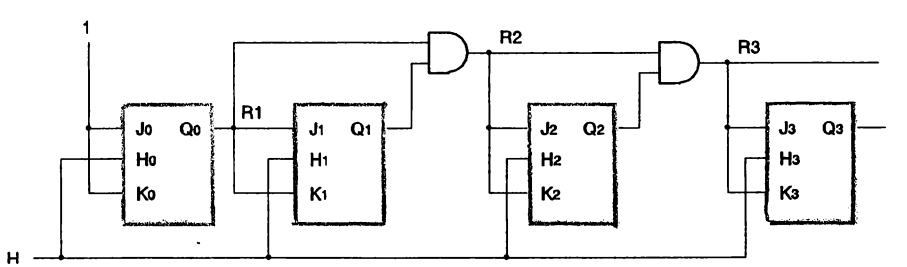



Fig. 2.26
Compteur binaire synchrone quatre étages.

Le même compteur peut être réalisé avec des bascules *D*. On a :

$$Q_n(t+1) = R_n \cdot \overline{Q_n} + \overline{R_n} \cdot Q_n$$

$$Q_n(t+1) = D$$

La figure 2.27 donne un exemple de réalisation de compteur binaire réalisé avec des bascules *D*.

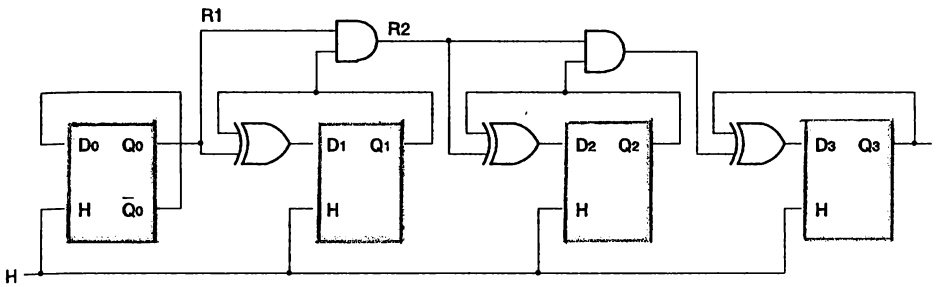



Fig. 2.27
Compteur binaire synchrone réalisé à partir de bascules *D*.

## 4 Synthèse des compteurs asynchrones

Pour ce premier exemple, on se replace dans le cas du compteur binaire à quatre étages.

On constate en analysant la table de vérité du compteur qu'un étage $Q_n$ change d'état si l'étage précédent passe de 1 à 0 et conserve son état dans les autres situations.

L'étage $Q_0$ qui n'a pas d'antécédent commute à chaque impulsion d'horloge. On peut donc décrire le comportement du compteur de la manière suivante :

- l'horloge est envoyée sur la première bascule *D*,
- le front descendant de la première bascule fait commuter la seconde,
- le front descendant de la deuxième bascule fait commuter la troisième,
- le front descendant de la troisième bascule fait commuter la quatrième.

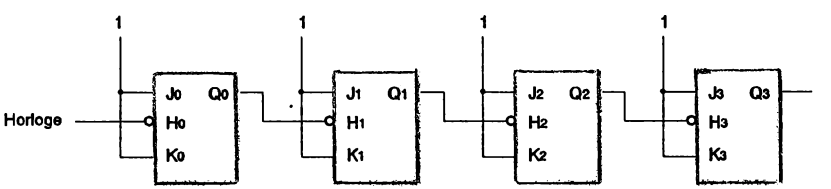



Fig. 2.28
Schéma de principe d'un compteur binaire asynchrone réalisé avec des bascules *JK*.

Prenons les bascules *JK* déclenchées sur front descendant. Pour la première bascule, $J_0$ et $K_0$ sont au niveau haut et l'horloge est envoyée sur l'entrée horloge.
Pour les étages suivants, la sortie $Q_{n-1}$ est appliquée directement sur l'entrée horloge de la bascule suivante $H_n$. La figure 2.28 donne le schéma correspondant.

Prenons maintenant les bascules de type *D* déclenchées sur le front montant. La première commutation est réalisée en envoyant l'horloge sur l'entrée horloge de la première bascule et en connectant $\overline{Q}_0$ sur $D_0$.

Lorsque l'horloge passe de 0 à 1 (front de montée), la bascule commute.

L'état de sortie est conservé lorsque le signal horloge repasse à zéro. À chaque front de montée, la bascule commute. Pour deux fronts de montée du signal horloge, il y a en sortie un front de montée et un front de descente.

La sortie de la première bascule est envoyée sur l'entrée horloge de la seconde bascule et le compteur asynchrone possède alors la configuration de la figure 2.29.

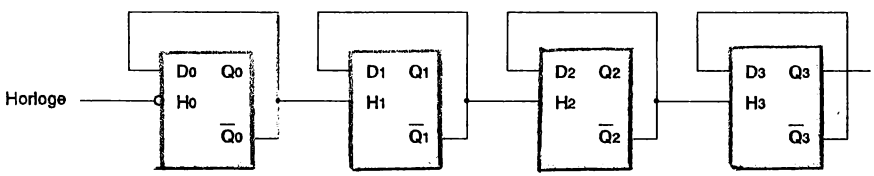



Fig. 2.29
Schéma de principe d'un compteur binaire asynchrone réalisé avec des bascules *D*.

## 5 Compteur modulo *N* ou diviseur par *N*

Comme dans l'exemple de la figure 2.25, diviseur par 3, on peut vouloir compter ou diviser en une base *N* différente de 2 avec des compteurs asynchrones.

Le cas le plus courant consiste à compter dans une base 10 puisque le système décimal est le plus familier.

La procédure décrite dans les lignes suivantes est applicable à une valeur quelconque de *N*.

Pour construire un compteur, on part d'une chaîne de *n* bascules en cascade telle que *n* soit le plus petit nombre pour lequel $2^n > N$.

On ajoute ensuite un dispositif qui effectue une remise à zéro des compteurs lorsque le contenu est égal à *N*.

La figure 2.30 représente un compteur par 10.

Lorsque les sorties $Q_3$ et $Q_1$ passent simultanément au niveau haut, les quatre bascules sont remises à zéro.

Dès que le contenu du compteur vaut 10, une impulsion négative apparaît en sortie de la porte NAND. Cette impulsion repositionne les compteurs à zéro.

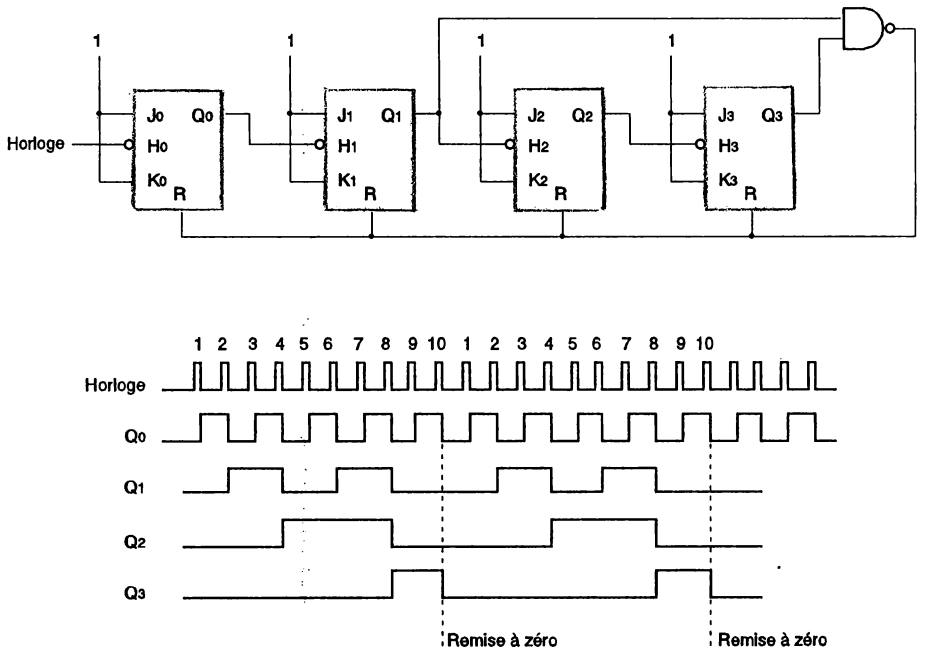



Fig. 2.30
Compteur par 10.

La sortie de la porte NAND peut être envoyée à l'entrée horloge d'un deuxième groupe de bascules identiques pour constituer une échelle de comptage de 0 à 99.

La procédure peut être étendue à un nombre quelconque de décades.

## 6 Exemples de décompteurs

Dans les exemples précédents, le cas des compteurs est traité. À chaque nouvelle impulsion d'horloge, le contenu des compteurs est incrémenté 3, 4, 5, 6, etc.

Pour certaines applications, il peut être intéressant de mettre en œuvre un décompteur. À chaque impulsion d'horloge, le contenu des compteurs est décrémenté d'une unité 6, 5, 4, 3, 2, etc.

La table de vérité du compteur s'établit suivant le tableau ci-après.

L'examen de la table de vérité montre que la bascule de rang zéro change d'état à chaque impulsion d'horloge.

Chaque bascule de rang *n* change d'état si et seulement si la bascule de rang *n*−1 passe de 0 à 1. Dans le cas contraire, la bascule de rang *n* ne change pas d'état.

Considérons par exemple le nombre 12 qui s'écrit 1100 en binaire. Le bit le moins significatif change systématiquement d'état. en conséquence la bascule 0 passe de 0 à 1.

| Valeur | $Q_3$ | $Q_2$ | $Q_1$ | $Q_0$ |
|---|---|---|---|---|
| 15 | 1 | 1 | 1 | 1 |
| 14 | 1 | 1 | 1 | 0 |
| 13 | 1 | 1 | 0 | 1 |
| 12 | 1 | 1 | 0 | 0 |
| 11 | 1 | 0 | 1 | 1 |
| 10 | 1 | 0 | 1 | 0 |
| 9 | 1 | 0 | 0 | 1 |
| 8 | 1 | 0 | 0 | 0 |
| 7 | 0 | 1 | 1 | 1 |
| 6 | 0 | 1 | 1 | 0 |
| 5 | 0 | 1 | 0 | 1 |
| 4 | 0 | 1 | 0 | 0 |
| 3 | 0 | 0 | 1 | 1 |
| 2 | 0 | 0 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 0 | 0 |

Ce qui implique, pour la bascule 1, un changement d'état. La bascule 1 passe de 0 à 1, ce qui implique un changement d'état, et la bascule 2 passe donc de 1 à 0.

Et finalement il n'y a pas de changement pour la bascule 3. On a en sortie du compteur la valeur 1011 qui correspond à 11 = 12 − 1. Le décompte a été effectué correctement.

Le schéma de la figure 2.31 représente une des configurations que l'on peut adopter pour transformer une échelle de comptage en comptage-décomptage.

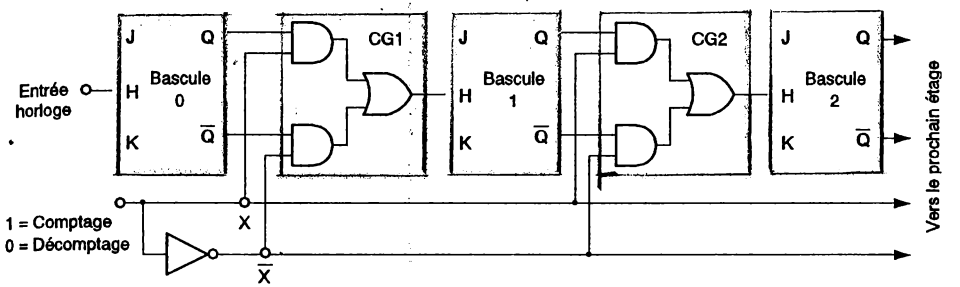



Fig. 2.31
Compteur-décompteur.

Si l'entrée de commande est au niveau haut, la cascade de bascule se comporte comme un compteur. Avec l'entrée de commande au niveau bas, on est en présence d'un décompteur.

La commande compteur-décompteur ne peut passer de 1 à 0 ou de 0 à 1 entre les impulsions d'horloge sans introduire un comptage parasite.

Pour remédier à cet inconvénient, on peut adopter la structure de la figure 2.32.

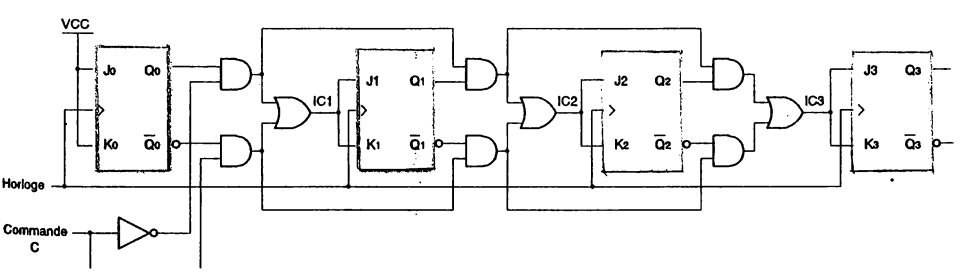



Fig. 2.32
Compteur-décompteur binaire.

© Dunod. La photocopie non autorisée est un délit.

## Applications des compteurs

Les applications des compteurs sont très nombreuses, nous nous bornerons ici à quelques exemples concrets.

### 1 Comptage direct

Dans de nombreux processus industriels, il est nécessaire d'effectuer un comptage. On peut facilement imaginer un système de comptage de véhicules permettant par exemple le calcul du débit en voitures par heure.

De la même manière, on peut compter le nombre de clients entrant et sortant d'un magasin.

Des capteurs appropriés transforment l'information en données électriques. Les informations électriques résultantes seront utilisées comme impulsions d'horloge et envoyées à l'**échelle de comptage**.

Si l'on souhaite que l'information soit exploitable et lisible immédiatement, on peut adopter le schéma de la figure 2.33 où les sorties de chaque décade sont exploitées pour piloter une circuiterie d'interface avec un afficheur sept segments.

Une circuiterie de logique combinatoire annexe peut être mise en place pour délivrer une impulsion ou un niveau lorsque le comptage atteint une certaine valeur *N*, le seuil d'alarme par exemple.

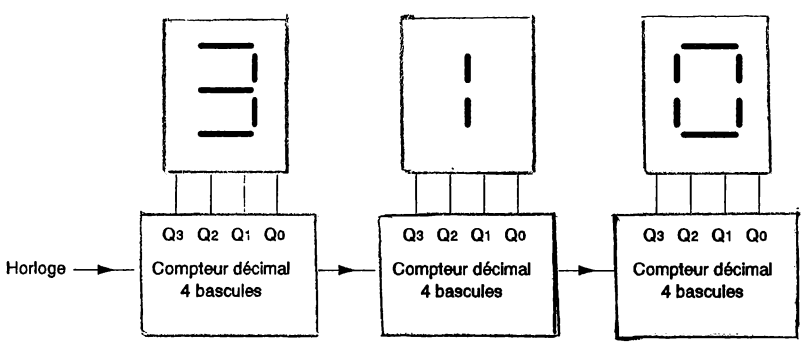



Fig. 2.33
Comptage direct et affichage.

## 2 Diviseur par *N*

À l'aide de *N* bascules *JK* ou *D*, il est possible de réaliser un compteur par *N* avec $2 \leq N \leq 2^n$, où *N* est un entier.

Les fréquences d'entrée et de sortie du compteur sont liées par la relation :

$$f_S = \frac{f_E}{N}$$

Le signal à la fréquence de sortie peut être prélevé directement sur la ligne de remise à zéro des bascules.

Dans ce cas, la fréquence de sortie est inférieure à la fréquence d'entrée. Le synoptique de la figure 2.34 donne une application extrêmement courante des compteurs : la boucle à verrouillage de phase PLL qui sera traitée au chapitre 8 de cet ouvrage.

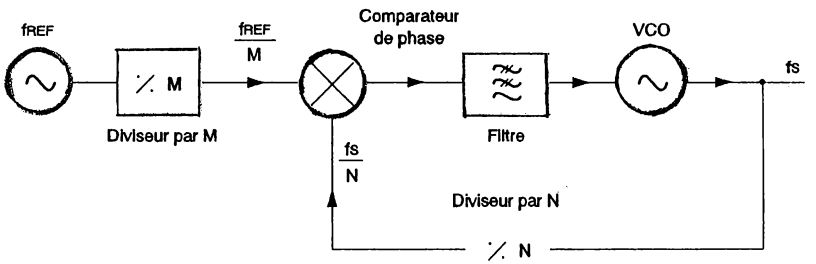



Fig. 2.34
Compteurs dans une boucle à verouillage de phase.

Le signal issu d'un oscillateur de référence est divisé par *M*. Le signal issu du VCO est divisé par *N*.

Le résultat de ces deux divisions $f_{REF}/M$ et $f_S/N$ est envoyé au comparateur de phase. Le signal d'erreur est filtré avant d'être envoyé au VCO.

La fréquence de sortie vaut : $f_S = \frac{N}{M} f_{REF}$

Le choix des diviseurs *M* et *N* détermine la fréquence de sortie $f_S$. Cette fréquence de sortie peut être plus importante que la fréquence de référence $f_{REF}$, ce qui n'était pas le cas dans la configuration précédente.

## 3 Mesure d'une fréquence

Une succession de compteurs, câblés conformément au schéma de la figure 2.35, permet la mesure et l'affichage direct de la fréquence d'un signal d'entrée.

Le signal de remise à zéro effectue une remise à zéro périodique de l'échelle de comptage. Dès que la RAZ est effectuée, l'échelle accumule les transitions du signal d'entrée. À la fin de la période de comptage, le résultat est transféré dans des bascules intermédiaires. Ce résultat reste affiché jusqu'à la prochaine mesure.

Dans le cas de la figure 2.35, la période de comptage est fixée à une seconde. L'impulsion de mémorisation est suivie d'une impulsion de remise à zéro et un nouveau cycle recommence.

Avec une **base de temps** d'une seconde, la valeur affichée est directement exprimée en cycles par seconde, soit en Hz. La résolution est de 1 Hz.

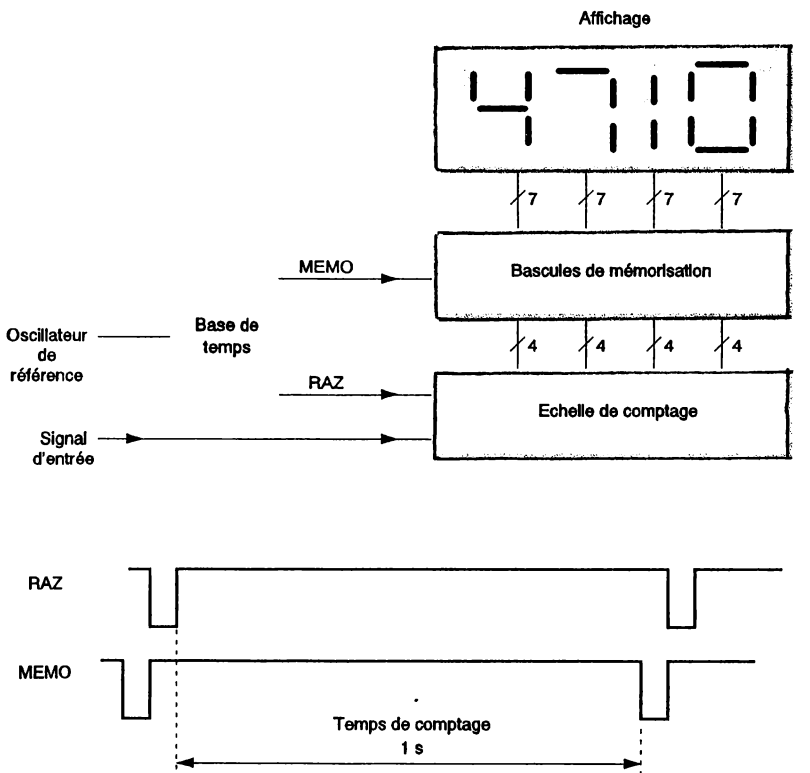



Fig. 2.35
Mesures de fréquence.

La valeur de la base de temps peut être modifiée pour augmenter la résolution, 10 secondes donnant une résolution de 0,1 Hz.

Des cadences plus rapides, 10 ou 100 mesures par secondes, se traduisent par une diminution de la résolution de 10 ou 100 Hz.

## 4 Mesure de temps

Un procédé similaire à celui de la mesure de fréquence autorise la mesure d'un temps ou d'une période. Supposons que l'on souhaite mesurer l'intervalle de temps séparant les deux impulsions de la figure 2.36.

Le front montant de la première impulsion agit sur une bascule qui se positionne de manière à autoriser le comptage d'une fréquence de référence : 1 MHz par exemple.

Le front montant de la seconde impulsion repositionne la bascule dans son état initial et le compteur est de nouveau inhibé. À la fin du cycle de comptage, le contenu du compteur est égal à : $N \cdot \frac{1}{f_{REF}}$.

La résolution est égale à $\frac{1}{f_{REF}}$ soit 1 µs dans l'exemple.

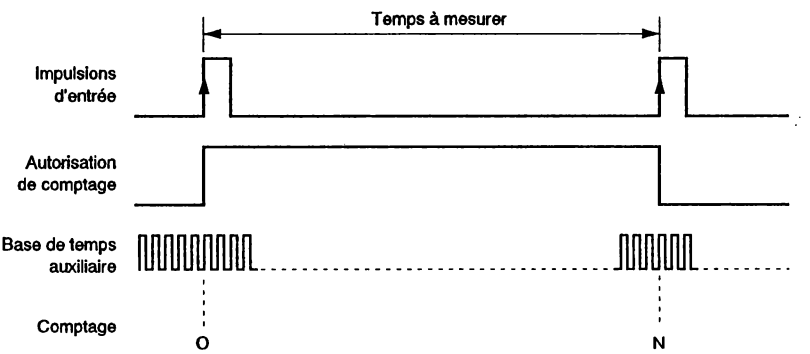



Fig. 2.36
Mesures de temps.

## 5 Mesure d'une distance

Dans les systèmes **RADAR** ou **SONAR**, la mesure du temps écoulé entre l'impulsion émise et l'impulsion reçue après réflexion - cible ou fond marin - permet de calculer la distance séparant l'émetteur et l'objet réfléchissant.

La mesure de la distance se résume à une mesure de temps.

## 6 Mesure de vitesse

La vitesse d'une cible, un véhicule par exemple, se mesure très facilement par effet **Doppler**.

Une onde de fréquence f est émise en direction de la cible. Si la cible est en mouvement, cette onde est réfléchie et renvoyée avec une fréquence $f + \Delta f$ :

$$\Delta f = \frac{2 \cdot v \cdot f}{c} \cos\theta$$

$f$ est la fréquence de l'onde émise,

$c$ est la vitesse de la lumière = $3\ 10^8$ m/s,

$\theta$ est l'angle entre la direction de la cible et la direction de l'onde.

Si $\theta = 90°$, il n'y a pas d'effet Doppler.

Dans la pratique, $f$ vaut 9 900 MHz ou 24 500 MHz.

Dans le récepteur de mesure, l'onde émise à la fréquence $f$ est mélangée avec l'onde reçue : $f + \Delta f$.

Le premier produit d'intermodulation, $(f + \Delta f) - f$, donne directement $\Delta f$.

La vitesse peut se déduire de la relation :

$$v = \frac{\Delta f \cdot c}{2 \cdot f \cdot \cos\theta}$$

Si $f$ = 9 900 MHz, $v$ = 50m/s (180 Km/h) et $\cos\theta = 1$. On obtient alors :

$$\Delta f = 3\ 300 \text{ Hz}.$$

Aux vitesses usuelles correspondent donc des fréquences inférieures à 5 000 Hz et mesurables simplement avec précision. Ces mesures de fréquences permettent donc la mesure immédiate de la vitesse.

## Description des systèmes de logique séquentielle en machines d'états

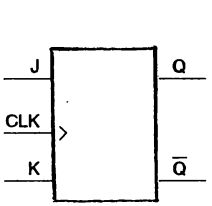
Une manière plus générale pour effectuer la synthèse des fonctions logiques consiste à déterminer les différents états possibles d'un système et à définir les conditions de passage d'un état à l'autre. L'identification des états possibles n'est pas toujours simple et les exemples qui suivent donneront quelques éléments sur la manière de poser le problème. Cette description en terme d'états est intéressante car les logiciels de synthèse acceptent cette description et permettent de générer plus ou moins automatiquement des portes logiques pour réaliser les fonctions.

### 1 Exemple de la bascule JK

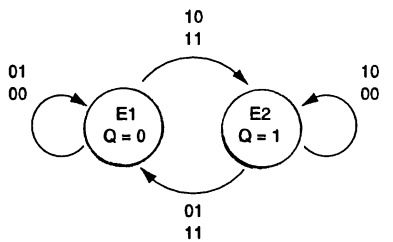
Les deux états possibles du système représenté figure 2.37.a sont :

État E1 : $Q = 0$ et $\overline{Q} = 1$

État E2 : $Q = 1$ et $\overline{Q} = 0$




*2.37.a*




*2.37.b*

Fig. 2.37
Étude d'une bascule *JK*.

Le passage d'un état à l'autre est donné par le fonctionnement de la bascule décrit précédemment. Il est donné par les états des entrées *J* et *K*. La condition est notée *JK*. La notation 10 signifie par exemple que *J* et *K* prennent les valeurs 1 et 0. Nous avons quatre combinaisons possibles pour les états de *J* et *K* et deux jeux de valeurs possibles pour les sorties. Il y a donc huit cas à examiner.

La figure 2.37.b représente graphiquement les passages d'un état à l'autre en fonction des conditions logiques. Elle se déduit de manière évidente des propriétés de la bascule. Il faut noter que les changements d'états sont synchronisés sur les fronts de l'hor-

loge. Cette particularité de la logique dite synchrone, la plus communément utilisée, n'a aucune incidence sur le graphe d'états. Dans une réalisation asynchrone, c'est-à-dire sans horloge, les changements d'états s'effectuent uniquement quand les variables changent mais le graphe des transitions reste le même. Les flèches du graphe indiquent le passage d'un état à un autre. Les conditions sont alors indiquées à côté de la flèche selon le symbolisme expliqué précédemment. Quand l'état ne change pas, la flèche part d'un état pour y revenir. Dans cet exemple, tous les états sont définis. Ce n'est pas toujours le cas. Dans un Set-Reset par exemple, l'état n'est pas défini quand les entrées Set et Reset sont simultanément à l'état haut. Dans ce cas, la représentation graphique n'est pas possible. Un deuxième exemple, plus complexe, illustre ce mode de représentation.

## 2 Exemple des feux de circulation

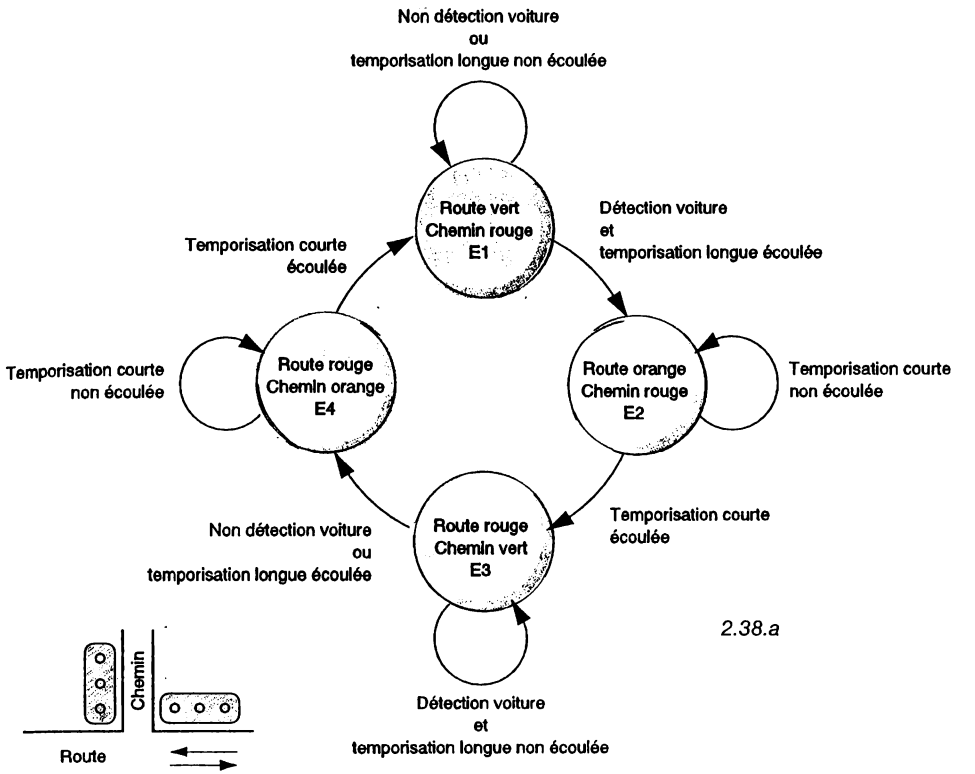
Cet exemple très classique dans la littérature électronique montre comment un problème de la vie quotidienne peut être décrit comme un système logique. Il donne également un exemple de machine d'états.

Une route principale croise un chemin. Comment régler les feux pour ne pas trop gêner la circulation ? Un capteur est placé sur le chemin pour détecter la présence d'un véhicule. Dans ce cas et seulement dans ce cas, le feu de la route principale passera au rouge. Pour ne pas trop gêner le trafic de la route principale, il est de plus nécessaire qu'un temps minimum soit écoulé. Ce temps est appelé temporisation longue. Les feux sont réglés de telle sorte que la sécurité soit assurée. Un feu vert ne peut correspondre qu'à un feu rouge sur l'autre route. La durée de l'orange est appelée temporisation courte. Les états possibles sont fonction des états des feux vert, rouge ou orange sur les deux voies. Les règles de sécurité réduisent ces états à quatre possibilités :

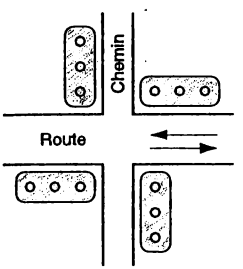
État 1 : route verte, chemin rouge
État 2 : route orange, chemin rouge
État 3 : route rouge, chemin orange
État 4 : route rouge, chemin vert

Les facteurs permettant de passer d'un état à l'autre sont : la détection d'une voiture sur le chemin, le dépassement ou non des valeurs de temporisation longue et courte.

Le schéma 2.38.a résume tous les cas possibles et s'explique de lui-même. Le schéma 2.38.b est identique à la seule différence que les phrases sont remplacées par des états vrai ou faux de variables logiques.




2.38.a



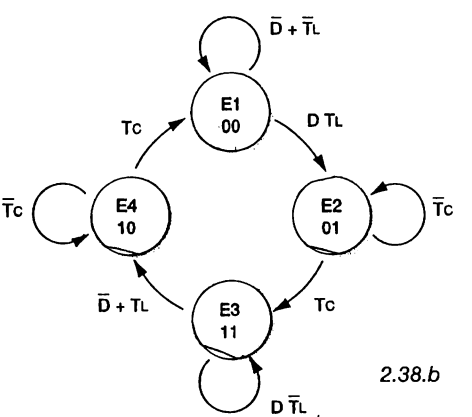

D : détection voiture sur le chemin

TL : temporisation longue écoulée

Tc : temporisation courte écoulée

Etat des feux :

E = (Q1, Q2)
E1 = (0, 0)
E2 = (0, 1)
E3 = (1, 1)
E4 = (1, 0)




2.38.b

Fig. 2.38
Feux de circulation.

La détection d'un véhicule sur le chemin implique que la variable $D$ soit égale à 1. La variable $T_L$ est égale à 1 quand la temporisation longue est écoulée. La variable $T_C$ correspond à la temporisation courte.

## 3 Synthèse des fonctions logiques

Deux classes de circuits sont définis : les logiques synchrones et les logiques asynchrones. Les systèmes synchrones sont les plus utilisés et les changements d'états s'effectuent en synchronisation avec les transitions de l'horloge.

Ce mode de fonctionnement garantit une plus grande sûreté que dans les systèmes asynchrones, en particulier vis-à-vis des parasites qui ont peu de chance d'être en coïncidence avec les transitions de l'horloge s'ils ne sont pas trop nombreux.

De plus, les systèmes asynchrones présentent souvent des comportements instables.

La figure 2.39 montre le schéma le plus général d'une logique séquentielle. Elle est formée de deux ensembles. Le premier est un ensemble de fonctions combinatoires, le second est un ensemble d'éléments mémoires permettant de garder le souvenir des états précédents. Ces éléments seront constitués de bascules RS ou de bascules JK selon le cas.

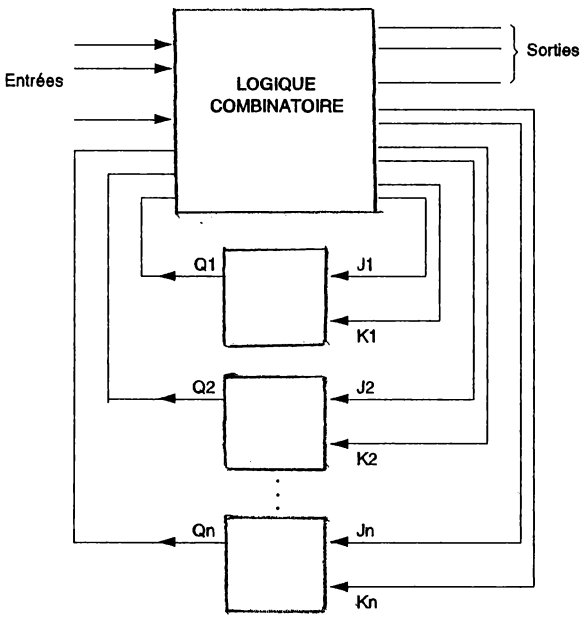



Fig. 2.39
Schéma général d'une logique séquentielle à $2^n$ états.

Le nombre d'éléments mémoires correspond au nombre d'états. Un système à quatre états pourra par exemple comprendre deux éléments mémoires pouvant représenter les quatre états possibles. En effet deux bits décrivent quatre états.

La figure 2.40.a décrit le schéma théorique de la logique associée au problème des feux et le schéma 2.40.b décrit la logique de manière précise. Le lecteur est invité à retrouver ce schéma à titre d'exercice. De nombreux logiciels permettent maintenant de générer un schéma à partir d'une description en machine d'états. Cette représentation présente donc un intérêt tout à fait particulier.

Pour trouver le schéma de la partie combinatoire du problème des feux, on peut tracer la table d'excitation du système. Elle est construite à partir du graphe d'états et elle indique les valeurs des entrées des deux bistables permettant de définir les quatre états possibles. Ces valeurs sont indiquées en fonction des états précédents et des valeurs des variables $D$, $T_c$ et $T_L$.

Il suffit de représenter les tables de Karnaugh des variables logiques $S_1$, $S_2$, $R_1$ et $R_2$, ainsi que celles des sorties $Q_1$ et $Q_2$. Les fonctions sont ensuite synthétisées par les méthodes classiques.

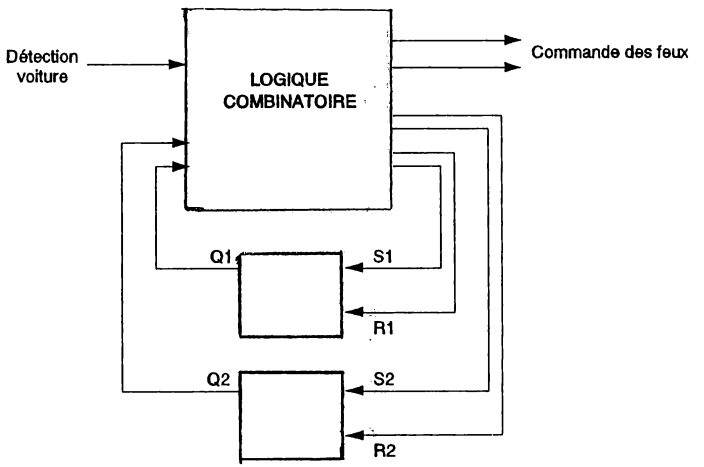



Fig. 2.40.a
Logique des feux de circulation.

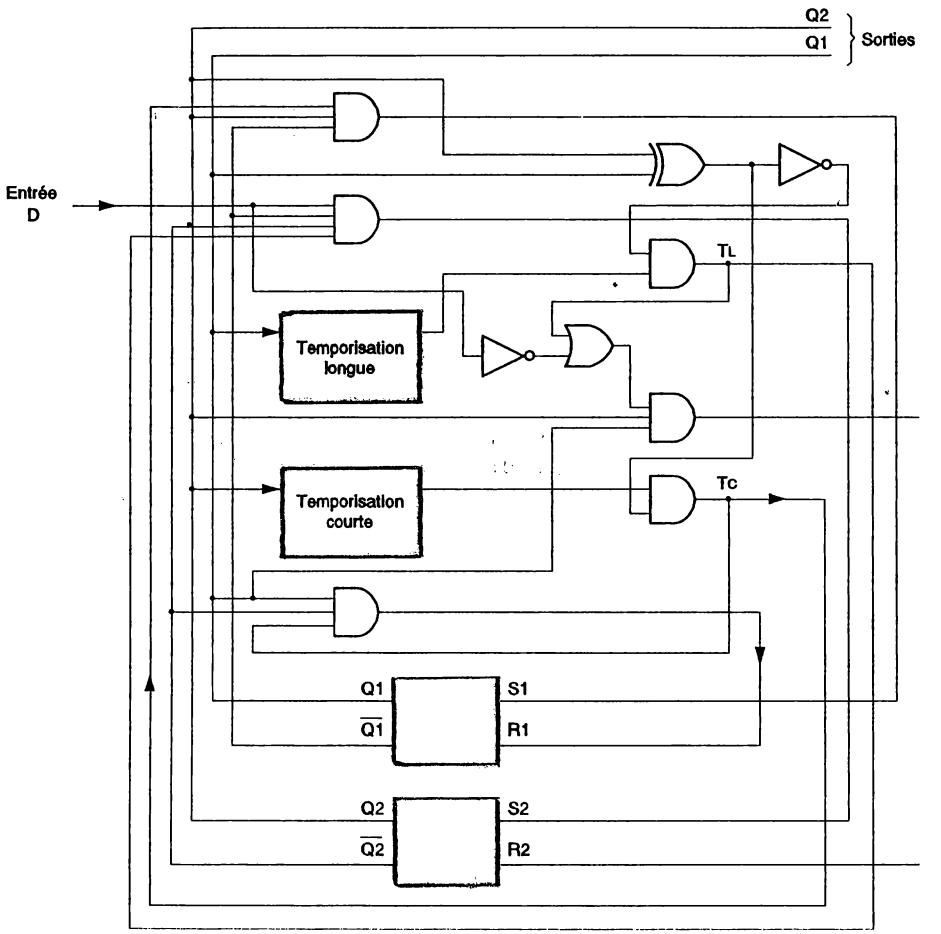



Fig. 2.40.b
Schéma détaillé de la logique des feux de circulation.

Le tableau 2.41 illustre le cas des feux de circulation.

| Etat | | | | | Etat suivant | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Q1 | Q2 | D | Tc | TL | Q1 | Q2 | S1 | R1 | S2 | R2 |
| 0 | 0 | 0 | X | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | X | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | X | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | X | 1 | 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | X | 0 | X | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | X | 1 | X | 1 | 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | X | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | X | 1 | 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | X | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | X | 1 | 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | X | 0 | X | 1 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | X | 1 | X | 0 | 0 | 0 | 1 | 0 | 0 |

$S1 = \overline{Q1}\ Q2\ Tc$

$R1 = Q1\ \overline{Q2}\ Tc$

$S2 = \overline{Q1}\ \overline{Q2}\ D\ TL$

$R2 = Q1\ Q2\ (\overline{D}\ \overline{TL} + D\ TL + \overline{D}\ \overline{TL}) = Q1\ Q2\ [\overline{D} + D\ TL]$

$= Q1\ Q2\ [\overline{D} + TL]$

Fig. 2.41

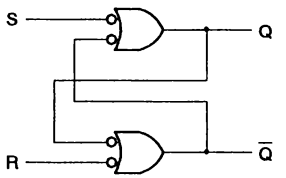
Table d'excitation des feux de circulation.

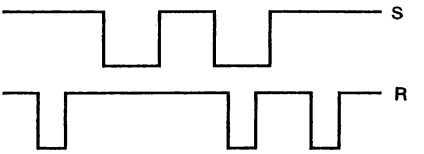
NB : on remarquera la formation de TC et TL à partir des états.

ENTRAÎNEMENT

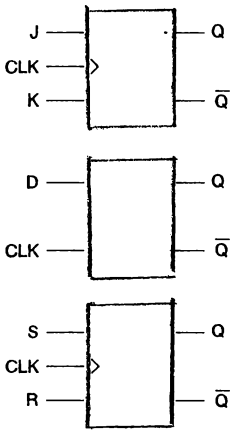
**1** Représentez sous forme d'un graphe le fonctionnement d'un compteur modulo 5.

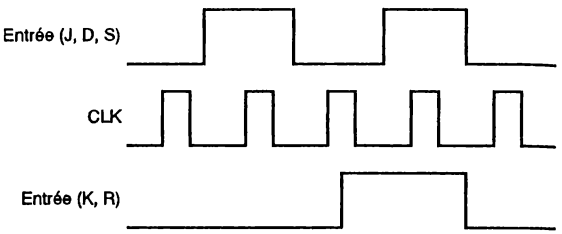
**2** Une bascule est représentée ci-dessous. Indiquez l'évolution des sorties dans le temps.




**3** Les mêmes signaux sont appliqués sur les entrées des trois bascules représentées ci-dessous. Représentez les sorties dans chaque cas.

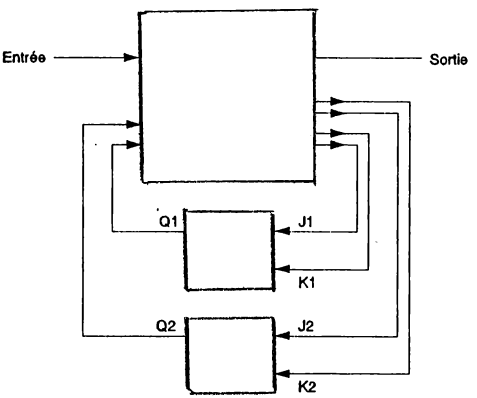



**4** Quelles sont les différences entre les compteurs synchrones et asynchrones ?

**5** Réalisez une horloge électronique à partir d'un signal alternatif de 60 Hz. Réalisez la même horloge à partir d'un signal à 50 Hz.

**6** Un compteur modulo 4 est représenté par un graphe d'états. Montrez qu'il peut être synthétisé par un schéma du type représenté à droite. Précisez la logique combinatoire.



ENTRAÎNEMENT

**7** Retrouvez le schéma du système des feux de circulation.

**8** Un intervalle de temps de 1 à 100 secondes doit être mesuré avec une précision de $10^{-3}$. Imaginez une logique séquentielle pour y arriver. Quelle doit-être la fréquence de l'horloge ?

**9** Un ascenseur dessert deux étages. Un détecteur indique si la porte de l'ascenseur est bien fermée. La priorité en cas de deux appels simultanés est donnée à l'étage le plus bas. Imaginez la logique de commande.

Chapitre 3

# LES FAMILLES LOGIQUES

**SOMMAIRE**


- Les propriétés des circuits logiques
- Les principales familles
- Les mémoires
- Entraînement


*Concevoir un système logique consiste à interconnecter des fonctions logiques de base : portes, bascules et registres. Les principes exposés précédemment sont mis en application pour déterminer les combinaisons les plus adaptées. Les outils de la CAO électronique permettent dans une large mesure d'automatiser le processus puisqu'il suffit d'écrire les équations logiques ou les tables de vérité pour obtenir le schéma électronique sous forme de portes élémentaires. Il est même possible de partir d'une description en terme de machine d'états ou d'une description comportementale, comme il sera vu à la fin de cet ouvrage.*
*Ce chapitre donne les règles électriques qu'il faut respecter et décrit les principales technologies disponibles. Une technologie est formée par un ensemble de composants de base (transistors et diodes) qu'il est possible d'interconnecter pour réaliser une fonction électronique complexe. Un important paragraphe est consacré à la description des mémoires électroniques qui sont parmi les circuits les plus courants de l'industrie électronique.*

## Les propriétés des circuits logiques

### 1 Marges de bruit et précautions à prendre

Une famille logique est un ensemble de produits permettant de réaliser des fonctions logiques. Ces produits peuvent être des circuits intégrés montés en boîtier ou des portes implantées dans une puce de silicium. Les niveaux logiques haut et bas ainsi que la vitesse de fonctionnement sont alors définis pour une famille donnée. La famille CMOS définit par exemple deux niveaux électriques 0 V et 3 V pour les deux états logiques. La fréquence maximale de fonctionnement est de l'ordre de quelques centaines de MHz pour les technologies les plus rapides. Les niveaux logiques

peuvent varier en particulier avec la température et de ce fait il faut considérer qu'un niveau logique n'est pas associé à une valeur précise mais à un ensemble de valeurs possibles.

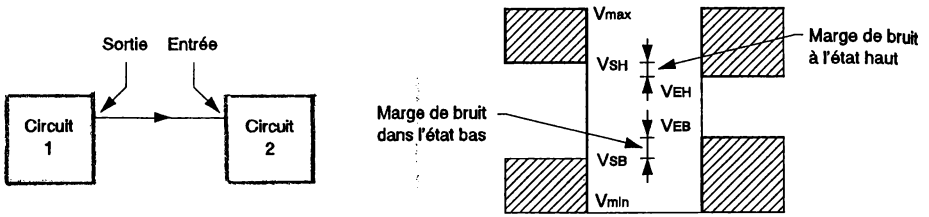



Fig. 3.1
Niveaux logiques et marges de bruits.

La figure 3.1 représente les valeurs possibles en sortie d'une porte logique.

Elles varient entre $V_{min}$ et $V_{SB}$ pour l'état bas et entre $V_{SH}$ et $V_{max}$ pour l'état haut. Une porte connectée en sortie de la première reconnaît un état logique bas si la tension en entrée est comprise entre $V_{min}$ et $V_{EB}$. Elle reconnaît un état haut si la tension en entrée est comprise entre $V_{EH}$ et $V_{max}$. Les valeurs maximales correspondent à une éventuelle destruction du circuit.

En définitive, la différence $V_{SH} - V_{EH}$ mesure la **marge de bruit** à l'état haut. La marge de bruit à l'état bas est définie de manière équivalente. Des variations de tension inférieures à cette valeur ne peuvent créer des états logiques non valides. La figure 3.2 montre l'effet d'une perturbation électromagnétique sur la sortie d'un inverseur. Dans le cas 3.2.a, la marge de bruit est dépassée, l'état de sortie est incertain. Dans le cas 3.2.b, le parasite est important et une erreur certaine est créée.

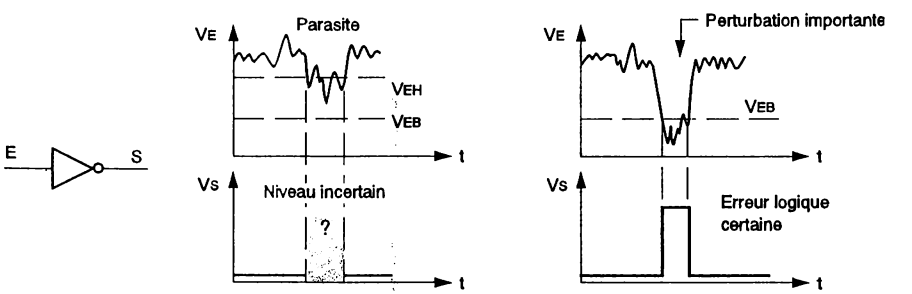



Fig. 3.2
Effet d'un parasite électrique.

Dans la pratique, il est nécessaire de respecter un certain nombre de règles élémentaires pour minimiser les risques de perturbations. Les deux règles principales sont le **découplage** des circuits et la réduction des longueurs d'interconnexions.

Pour découpler un circuit, il faut placer un condensateur de valeur élevée entre le point d'alimentation du circuit et la référence de potentiel également appelée **masse**.

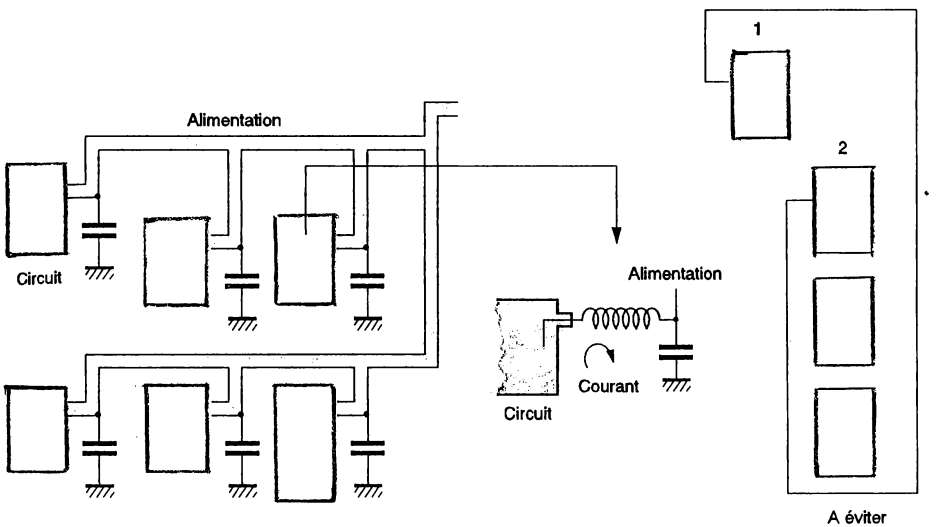



Fig. 3.3
Découplage et interconnexion des circuits.

La figure 3.3 illustre la notion de découplage. La référence de potentiel est un plan métallisé pour une carte. Dans un circuit intégré, c'est un plan correspondant au substrat. La connexion doit être la plus courte possible pour réduire la valeur de l'inductance du fil utilisé. La présence d'une inductance dans le circuit de découplage aurait pour effet de créer un circuit résonnant série, propre à créer des oscillations ou des instabilités. Le rôle précis du condensateur de découplage peut s'expliquer en considérant les courants variables créés par les transitions logiques. Ces courants ont un contenu spectral important dans le domaine des fréquences élevées correspondant à des transitions rapides. Quand le découplage est de bonne qualité, l'inductance série est de faible valeur. Le circuit de découplage est alors équivalent à un court-circuit et le courant variable associé à la transition est dévié vers la masse sans perturber les autres circuits.

La deuxième règle importante est de réduire les longueurs des interconnexions entre circuits ou entre portes. Cette règle permet de minimiser l'effet des réflexions de l'impulsion logique en extrémité de la piste conductrice.

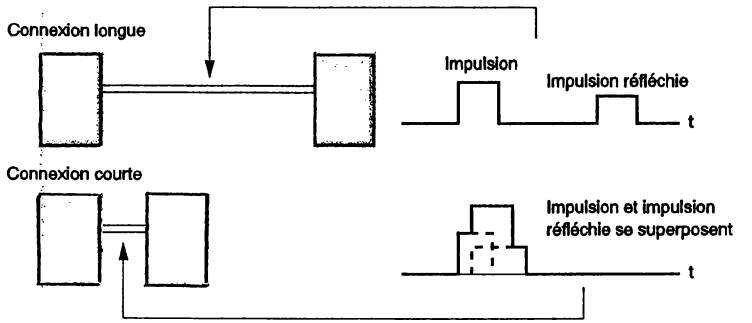



Fig. 3.4
Influence de la longueur d'interconnexion.

La figure 3.4 illustre cet effet dans le cas d'une piste courte et dans le cas d'une piste longue. Dans le premier cas, l'impulsion réfléchie se superpose instantanément à l'impulsion fournie et aucun effet n'est mesurable. Dans le deuxième cas, l'impulsion ou les impulsions réfléchies sont clairement séparées du temps d'aller et retour de l'impulsion dans la ligne formée par la piste. Des états logiques erronés peuvent apparaître.

Dans un circuit intégré, les distances entre portes sont courtes, au maximum de quelques millimètres et de ce point de vue, la situation est favorable. Par contre, les distances étant faibles, les capacités parasites de couplage sont élevées entre les pistes conductrices et les risques de perturbation sont importants. Pour des fréquences de fonctionnement élevées, il est parfois nécessaire d'adapter les lignes de transmission. Dans ce cas, le circuit récepteur doit présenter une impédance d'entrée égale à l'impédance caractéristique de la ligne.

Cette technique est utilisée en technologie ECL pour des fréquences de fonctionnement supérieures à 50 MHz environ et des longueurs de connexion supérieures à quelques centimètres.

## 2 Temps de commutation et temps de propagation

Les transistors réalisant les fonctions logiques ne sont pas parfaits et quand une impulsion logique est appliquée en entrée, elle est à la fois retardée et déformée. La figure 3.5 représente le cas d'un inverseur. L'impulsion d'entrée est supposée idéale, les temps de montée et temps de descente sont alors nuls. L'impulsion en sortie est retardée d'un temps *tp* appelé temps de

propagation. De plus, elle présente un temps de montée *tm* non nul et un temps de descente *td* non nul également. Le même phénomène peut s'observer en sortie d'une fonction logique quelconque. Cet effet est de type intégrateur comme il a été vu dans le chapitre consacré au calcul des circuits. Il est lié à la bande passante limitée des transistors de la technologie considérée.

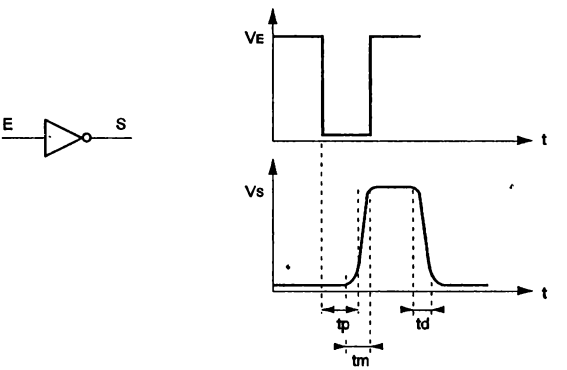



Fig. 3.5
Temps de propagation et temps de commutation.

Dans un circuit tel que celui représenté figure 3.6, le retard entre entrée et sortie est la somme des retards créés dans chaque porte. Les technologies actuelles offrent des retards par porte aussi faibles que 100 picosecondes pour les produits les plus rapides. Minimiser les temps de propagation conduit à augmenter la fréquence de fonctionnement possible du système. Pour en être convaincu, il suffit d'imaginer un rebouclage entre une sortie et une entrée. Il est alors nécessaire d'attendre la formation de l'impulsion logique de sortie avant de pouvoir changer l'état logique d'entrée.

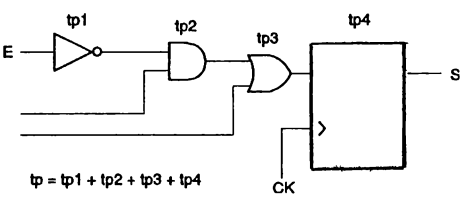



Fig. 3.6
Retard dans un circuit.

## 3 Temps de *hold* et de *set-up*

Dans la logique synchrone, les sorties logiques des circuits apparaissent en phase avec un signal de synchronisation distribué à tous les circuits : l'horloge.

C'est un signal périodique, représenté figure 3.7. Les différentes sorties sont donc inspectées par le signal d'horloge, et au moment de la transition, l'état logique correspondant à la fonction est pris en compte et maintenu pendant la durée de la période. Ce mode de fonctionnement quasi-universel permet de réduire les aléas des circuits. Dans un fonctionnement asynchrone, c'est-à-dire sans horloge de synchronisation, un parasite fugitif suffit pour changer un état logique et donc perturbe le fonctionnement du système.

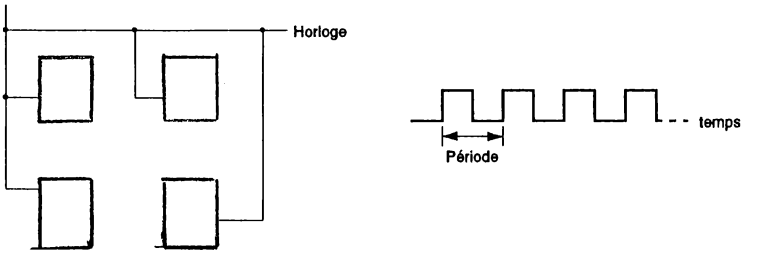



Fig. 3.7
Signal d'horloge.

Dans la logique synchrone, il est nécessaire que les états logiques d'entrée soient définis au moment de la transition. Dans le cas de la bascule « D », représentée figure 3.8, la donnée est prise en compte sous deux conditions :
- le niveau logique d'entrée doit être établi depuis un temps minimum avant la transition d'horloge. Ce temps minimum est appelé **temps de *set-up.*** Sa valeur est de l'ordre de quelques nanosecondes pour les technologies actuelles,
- le niveau logique d'entrée doit être maintenu un temps minimum après la transition de l'horloge. Ce temps est appelé **temps de *hold***. Sa valeur est parfois inférieure à la nanoseconde.

La même définition s'applique à des circuits plus complexes comme les registres ou les mémoires.

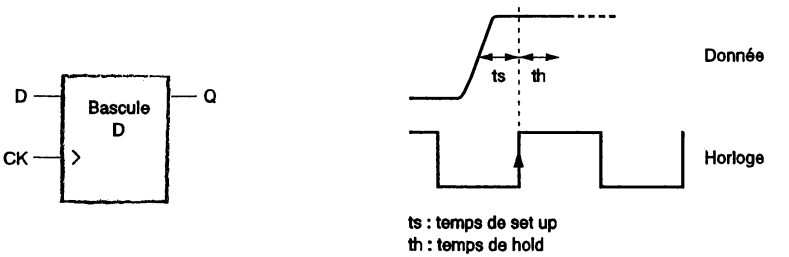



Fig. 3.8
Temps de *hold* et de *set-up*.

## 4 Entrance et sortance : *Fan-in* et *Fan-out*

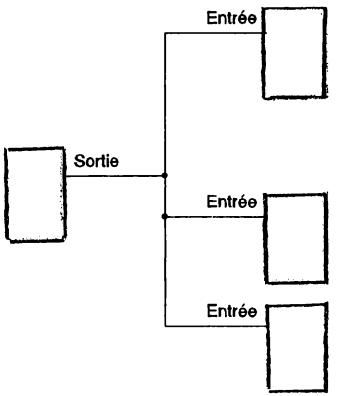



Fig. 3.9
Sortance d'un circuit.

Dans de nombreux cas pratiques, il est nécessaire de relier une sortie logique à plusieurs entrées. Les exemples donnés dans les chapitres précédents en témoignent. La même expression logique peut par exemple être utilisée dans plusieurs fonctions, comme le montre la figure 3.9. La question se pose alors de déterminer le nombre maximum d'entrées qu'il est possible de relier à une même sortie. La réponse est donnée par une caractéristique de la technologie appelée **Fan-out** ou **sortance**. La figure 3.10 permet de comprendre que ce chiffre est lié à la valeur des courants consommés par les entrées et à la valeur de la capacité électrique du circuit d'entrée. Plus les circuits sont nombreux, plus le courant débité par le circuit de commande est important. Plus les circuits sont nombreux, plus la capacité *C* vue par le circuit de commande est élevée. Dans ce cas, une constante de temps *RC* intègre le signal et la fréquence maximale de fonctionnement est réduite.

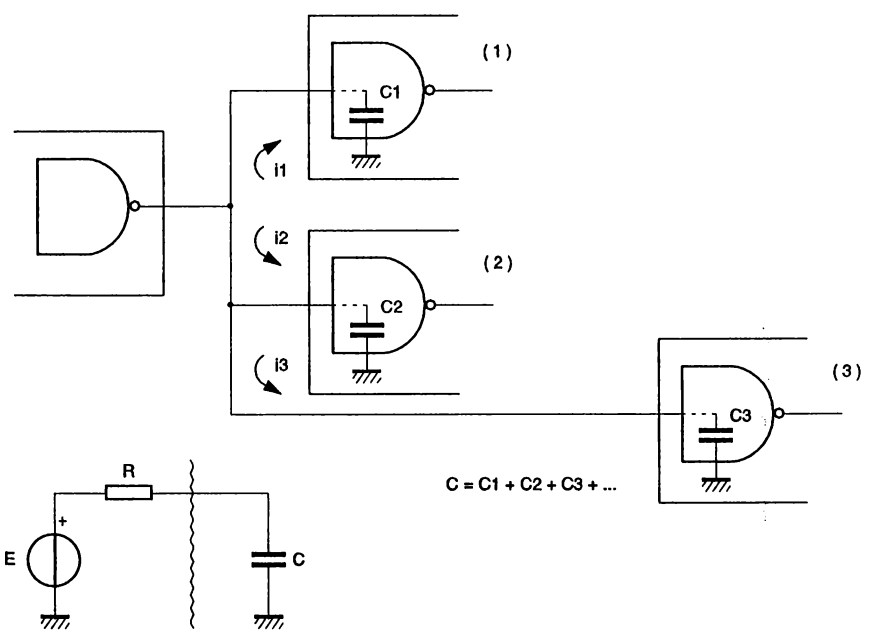



Fig. 3.10
Sortance d'un circuit.

De la même manière, il est possible de définir l'entrance d'un circuit. C'est le nombre maximum de sorties qu'il est possible de connecter sur une même entrée. Ce montage n'est pas possible pour tous les circuits d'une technologie donnée. Il est réservé aux circuits ayant un étage de sortie de type collecteur ouvert. Ce type de montage est appelé « **ET câblé** » dans la littérature électronique.

## 5 Fonctions et réalisation d'un *ET* cablé

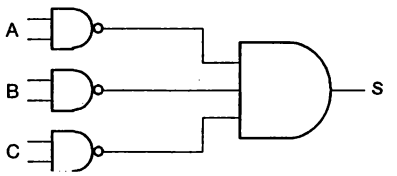



Fig. 3.11
*ET* câblé avec une porte.

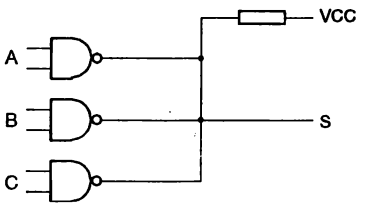
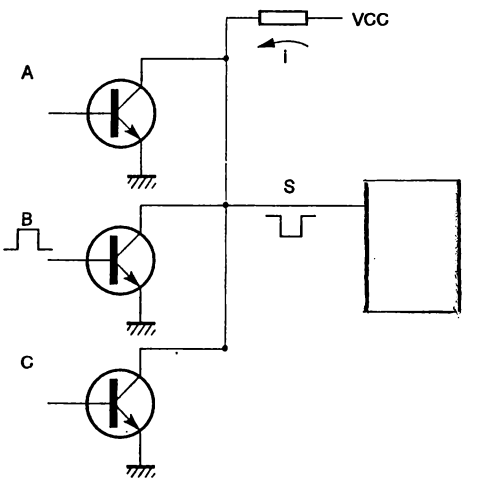



Fig. 3.12
*ET* câblé.

Les figures 3.11 et 3.12 illustrent deux manières de réaliser une fonction *ET* entre différentes sorties. La première, illustrée en 3.11 consiste à utiliser un circuit spécial. La deuxième, plus économique, consiste à réaliser le montage indiqué figure 3.12. Le schéma détaillé est représenté figure 3.13. Il montre que les sorties sont des montages émetteur-commun ou collecteurs ouverts avec une seule résistance de charge. La sortie du dispositif n'est à l'état haut que si tous les transistors de sortie sont non passants. Toutes les tensions de base sont alors à l'état bas. La fonction *OU* inversée est alors réalisée.




**Si un des transistors conduit, alors S est à l'état bas. (B sur la figure)**

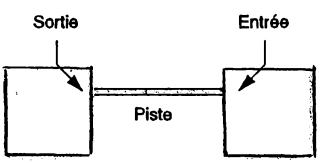
$S = \overline{A + B + C} = \overline{A}\,\overline{B}\,\overline{C}$

Fig. 3.13
*ET* câblé avec les collecteurs ouverts.

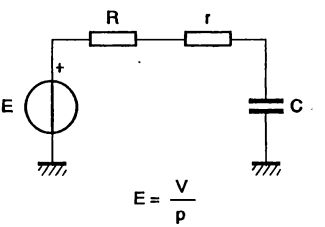
## 6 Consommation statique et dynamique

Cette notion est très importante car le niveau de complexité des circuits électroniques actuels est tel que la consommation électrique est souvent le facteur le plus contraignant. La consommation est soit statique soit dynamique.

La notion de **consommation statique** est la plus simple, elle correspond simplement à la consommation dans les deux états logiques d'une porte. Si les courants débités sont $I_B$ et $I_H$, la puissance dissipée par le circuit est alors une moyenne entre $V_{CC}I_B$ et $V_{CC}I_H$ (la tension $V_{CC}$ étant la tension d'alimentation). Quand les portes sont réalisées dans une technologie bipolaire, la consommation statique peut être importante. Dans la technologie CMOS, d'utilisation courante actuellement, la consommation statique est négligeable puisque les courants débités sont très faibles. Il ne faudrait pas en déduire hâtivement que la logique CMOS ne pose aucun problème de consommation. Il faut alors tenir compte de la **consommation dite dynamique** (figure 3.14).




*3.14.b*




*3.14.a*

Fig. 3.14
Calcul de la consommation dynamique.

Le schéma de la figure 3.14.a permet de montrer l'origine de la consommation dynamique. Deux portes sont interconnectées par une piste de résistance *r*. On suppose que les deux portes logiques sont représentées par le schéma 3.14.b. La porte de commande est équivalente à un générateur de tension de résistance interne *R*. La porte commandée est équivalente à une capacité *C*.

Cette représentation est bien adaptée à la logique CMOS. La résistance d'entrée est quasi infinie. La résistance de sortie est assez faible et la capacité d'entrée est de l'ordre de 5 pF. Cette valeur correspond à la capacité d'entrée d'un circuit. Quand on réunit deux portes sur une même puce, elle est de valeur beaucoup plus faible ($10^{-2}$ pF).

Il est alors facile de montrer que dans une transition d'amplitude $V$, l'énergie dissipée dans la résistance $R + r$ est donnée par la relation :

$$W = \frac{1}{2}CV^2$$

On constate que cette valeur ne dépend pas des résistances $r$ et $R$. Ce résultat est proposé comme exercice d'application. Si on considère l'énergie dissipée lors de la transition opposée, on obtient également la même valeur. L'énergie dissipée dans la transmission d'une impulsion logique d'amplitude $V$ est donc :

$$W = CV^2$$

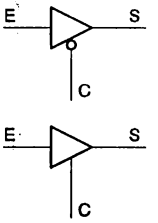
La puissance dissipée par un circuit dans lequel $N$ portes commutent à une fréquence $f$ est donc :

$$P = NfCV^2$$

Cette formule indique clairement que la consommation augmente linéairement avec la fréquence de fonctionnement et avec la complexité du circuit et qu'elle varie fortement avec la tension d'alimentation. Les circuits basse consommation utilisent donc une tension d'alimentation de 3 V et même de 1,5 V dans des applications particulières comme l'horlogerie ou le jouet. Un circuit comportant $10^6$ portes commutant à la fréquence de 100 MHz dissipe une puissance dynamique de 9 W avec une capacité par porte aussi faible que $10^{-14}$ F et une tension d'alimentation de 3 V.

## Logique à trois états

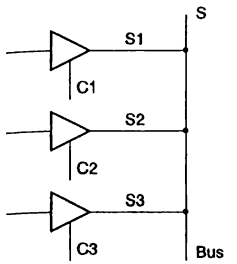
Dans un certain nombre de cas, il est nécessaire que la sortie d'un circuit logique soit à haute impédance, c'est-à-dire présente une impédance de sortie élevée. Des circuits spéciaux ont été développés à cet effet. Ils sont appelés circuits **trois états**. Ces trois états sont : l'état haut, l'état bas et l'état à haute impédance. La figure 3.15 donne le schéma de principe d'un tel circuit. La tension de commande permet d'isoler électriquement le circuit.



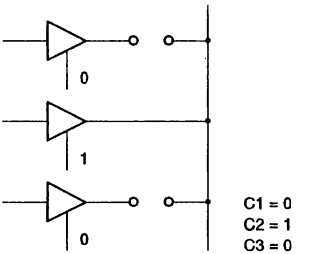

C = 0 S = E
C = 1 Haute impédance en sortie

C = 0 Haute impédance en sortie
C = 1 S = E

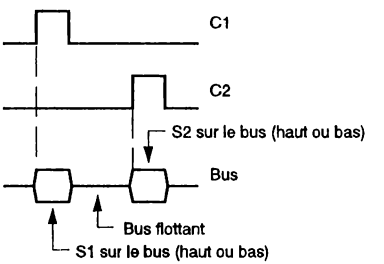
Fig. 3.15
Circuits à trois états.




*3.16.a*




*3.16.b*




*3.16.c*

**Fig. 3.16**
Circuits trois états et bus.

Quand la commande est dans l'état contraire, le circuit transmet alors l'état logique d'entrée mais dans ce cas, il présente une impédance de valeur non infinie et parfois même de valeur assez faible.

La figure 3.16 illustre l'application type des circuits trois états. Il s'agit de relier différents circuits sur une même ligne électrique. La ligne électrique peut transporter les états logiques en provenance des divers circuits 1, 2, 3, etc. Les circuits sont alors dits reliés par un **bus** (figure 3.16.a). Les commandes $C_1$ , $C_2$ , $C_3$ , etc., permettent de sélectionner dans le temps le circuit concerné et permettent également de placer les circuits dans un état à haute impédance quand ils ne sont pas concernés par l'échange de données. La figure 3.16.c représente la séquence temporelle des signaux.

Il faut maintenant expliquer un symbole utilisé couramment dans la littérature électronique et représenté figure 3.16.c. Les états logiques étant indifféremment haut ou bas, le symbole logique représente simutanément les deux formes électriques possibles.

Quand les circuits reliés au bus ne sont pas actifs, ils sont alors dans le troisième état à haute impédance. Le bus est alors dit **flottant**. Un signal extérieur peut fixer son niveau électrique à une valeur quelconque.

## Les familles logiques

Il ne s'agit pas dans ce paragraphe de décrire en détail les circuits logiques dans les différentes technologies. Des ouvrages spécialisés sont consacrés à ce sujet. De plus, les connaissances nécessaires pour mettre en œuvre des applications à partir de ces circuits sont somme toute assez réduites. L'objectif de ce paragraphe est de rassembler les connaissances requises pour utiliser les principales technologies actuelles. La conception d'un ensemble logique peut se faire sous la forme d'une carte électronique et il s'agit de relier différents circuits intégrés en respectant les règles de l'art. Elle peut également se faire sous forme d'un circuit intégré ou **ASIC**, *Application Specific Integrated Circuit*. Il faut alors relier entre eux les transistors fabriqués dans la puce. Les règles d'interconnexion sont voisines dans les principes.

### 1 Logique TTL (*Transistor Transistor Logic*)

Historiquement, c'est la technologie la plus ancienne. Elle est formée de transistors bipolaires et de diodes. Le niveau bas est de l'ordre de 0,2 V et le niveau haut de l'ordre de 3,4 V. Elle a peu à peu fait place à la technologie CMOS mais elle est encore utilisée, en particulier dans les interfaces. Cette technologie permet de réaliser facilement des *ET* câblés et des circuits à trois états. Elle comporte un certain nombre de sous-familles : TTL rapide, TTL basse consommation. Le tableau 3.17 classe les sous-familles en fonction de la consommation statique par porte et du temps de propagation par porte. Les familles 74ALSXX et 74FXX sont les versions les plus modernes et donc les plus performantes. Les symboles XX sont des chiffres permettant d'identifier les fonctions logiques particulières.

| | Symbole | Temps de propagation par porte en ns | Consommation statique par porte en mW |
|---|---|---|---|
| Standard | 74XX | 10 | 10 |
| Basse consommation | 74LXX | 30 | 1 |
| Rapide | 74HXX | 16 | 30 |
| Shottky | 74SXX | 3 | 20 |
| Shottky avancée | 74ASXX | 1 | 8 |
| Shottky basse consommation | 74LSXX | 10 | 2 |
| Shottky basse consommation avancée | 74ALSXX | 4 | 1 |
| Rapide | 74FXX | 2 | 4 |

Fig. 3.17
Les familles logiques TTL.

La figure 3.18 représente quelques circuits intégrés classiques dans cette technologie. Le lecteur retrouvera sans peine des fonctions décrites dans les chapitres précédents.

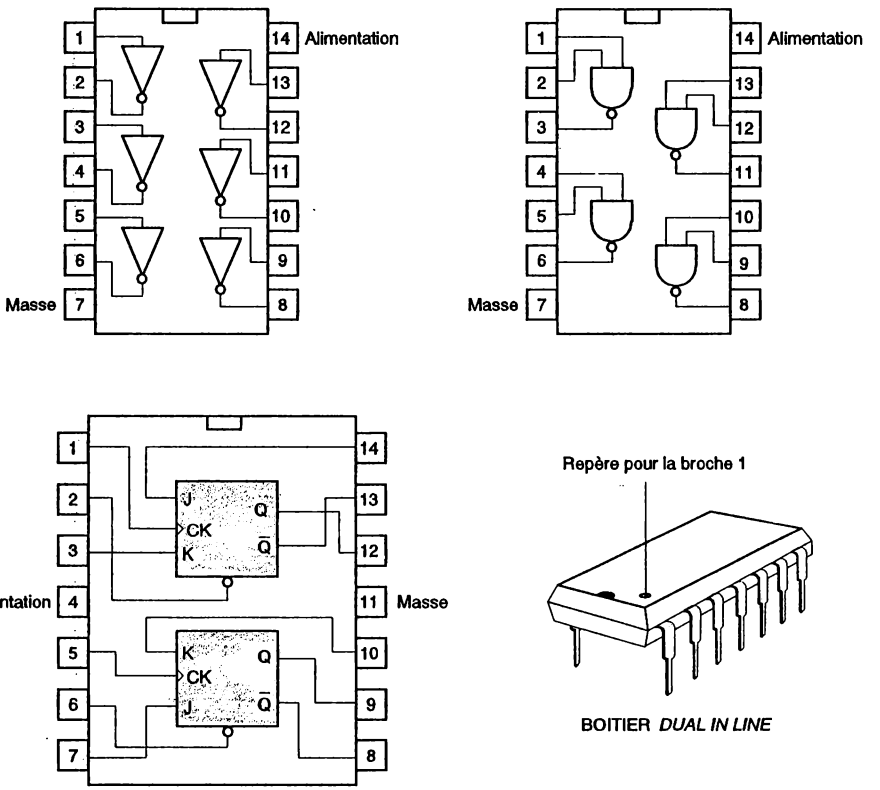



Fig. 3.18
Exemples de circuits.

La figure 3.19 représente une fonction TTL standard, la porte NAND. La description détaillée de son fonctionnement est donnée pour les lecteurs plus précisément intéressés par la technologie. Elle peut être oubliée dans une première lecture.

Cette fonction utilise un transistor *T1* particulier puisqu'il comporte deux émetteurs en parallèle. La représentation de ce transistor sous forme de trois diodes permet de mieux comprendre son fonctionnement. Si les deux entrées *E1* et *E2* sont à l'état haut, le collecteur du transistor *T1* est à un potentiel haut et le transistor *T2* est mis en conduction. Cette conduction entraîne la saturation du transistor *T4*.

La tension de sortie est alors la tension de saturation du transistor *T4* soit 0,2 V environ et la sortie est à l'état bas. A l'opposé, le transistor *T3* est non conducteur. La diode D conforte ce comportement. Si l'une des entrées au moins est à l'état bas, alors la base de *T2* est à l'état bas et le transistor *T2* est non passant. Dans ce cas, *T4* est bloqué et *T3* est conducteur. La tension en sortie est dans ce cas donnée par la relation :

$$V_S = V_{CC} - V_D - V_{BE3}$$

La tension aux bornes de la résistance de 1,6 kΩ est nulle puisque *T2* est bloqué. La tension de sortie est donc voisine de 3,6 V puisque la tension d'alimentation est de 5 V et que les deux tensions de diode sont de l'ordre de 700 mV. Les diodes placées en entrée protègent le circuit contre des pointes de tension négatives.

Les versions **Schottky** utilisent des transistors et des diodes particulières permettant d'éviter une augmentation du temps de propagation consécutif à la saturation des transistors. Les dispositifs dits Schottky éliminent les phénomènes de saturation et permettent de réduire les temps de propagation donc d'augmenter la fréquence maximale de fonctionnement. La figure 3.20 représente une version légèrement différente du circuit NAND. La sortie est dite à collecteur ouvert. Le fonctionnement du circuit est très voisin de celui de la figure 3.19. Comme il a été vu précédemment, cette architecture permet de réaliser une fonction *ET* câblée.

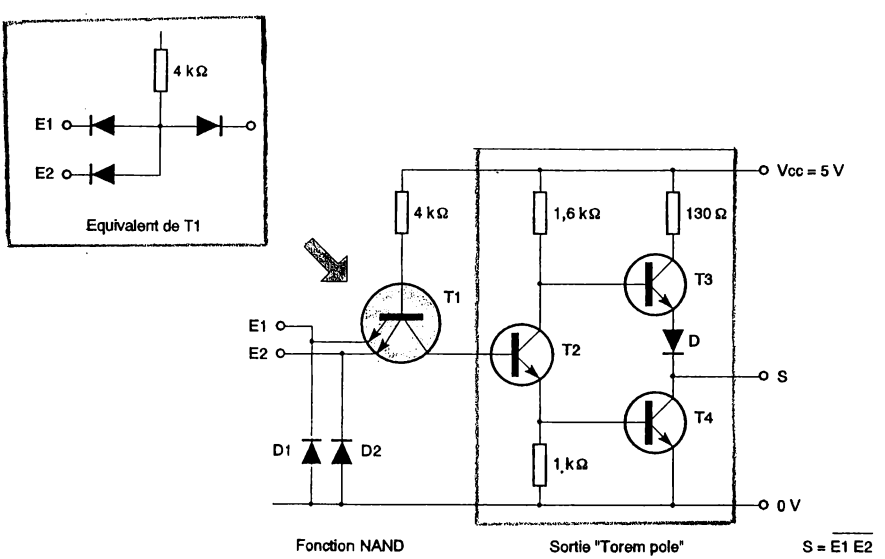



Fig. 3.19
Porte NAND TTL.

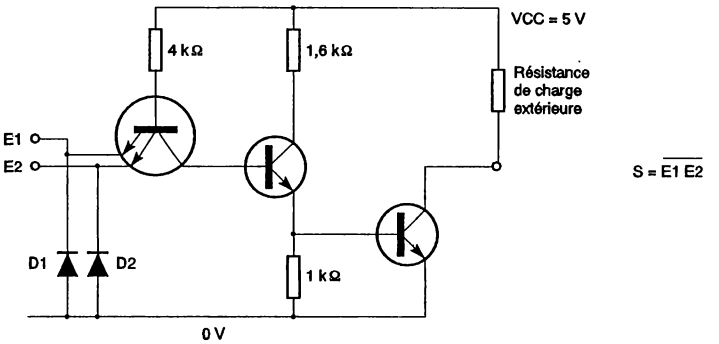



Fig. 3.20
Porte NAND avec collecteur ouvert.

Les autres fonctions logiques peuvent s'obtenir assez facilement à partir de ces deux architectures et de quelques inverseurs logiques. L'établissement de ce résultat est proposé à titre d'exercice.

Pour terminer cette brève description de la logique TTL, il convient de donner les principales règles d'interconnexion des circuits ou des fonctions. Le lecteur doit être conscient du fait que ces règles sont très générales et que les règles précises sont données dans les livres d'applications des constructeurs relatifs à chaque sous-famille.

Les quelques recommandations générales sont les suivantes :
– les circuits se connectent de sortie à entrée comme l'indique la figure 3.21. Il est possible de relier une même sortie à une dizaine d'entrées environ. On dit que le fan-out est de 10 ;

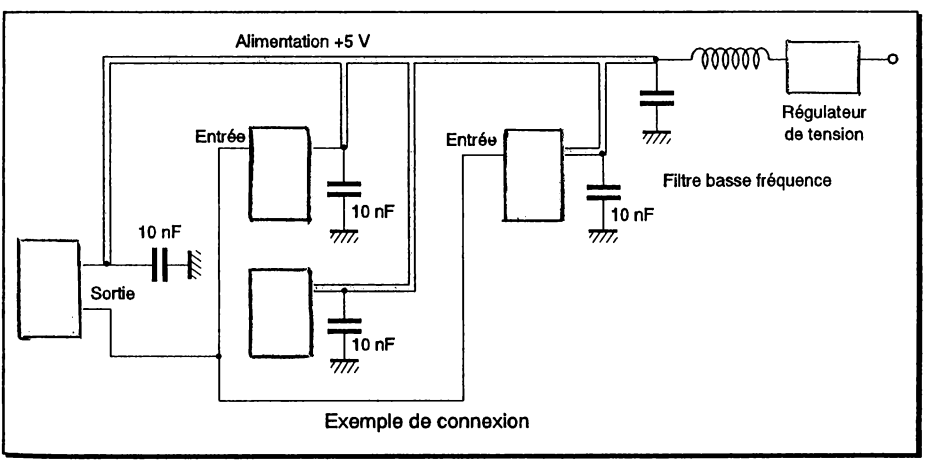



Fig. 3.21
Carte logique TTL.

– les circuits sont alimentés en 5 V en général et chaque circuit doit être soigneusement découplé par un condensateur de 100 nF environ, relié au plan de masse par le chemin le plus court possible. L'alimentation générale de la carte doit comporter un filtre pour éliminer les composantes résiduelles basse fréquence. Les régulateurs de tension, étudiés dans le chapitre 6, sont assez communément utilisés ;
– certaines entrées des circuits logiques peuvent être non utilisées. C'est par exemple le cas d'un *OU* logique à 6 entrées dont on n'utilise que quatre d'entre elles.
La question se pose alors : doit-on laisser les entrées non utilisées non connectées ou doit-on fixer leur potentiel ? Dans le cas de la fonction *OU*, il est possible de fixer les entrées non utilisées dans le niveau logique bas en les reliant à la masse. Les entrées, devant être fixées au niveau logique haut, pourront être reliées directement au 5 V par une résistance de quelques kΩ. 4,7 K
Dans tous les cas, il est déconseillé de laisser flotter les entrées bien que, en théorie, elles soient alors dans l'état logique haut ;
– le montage en *ET* câblé a été décrit précédemment. Il nécessite d'utiliser des sorties de type collecteur ouvert et une résistance extérieure de charge doit être connectée ;
– les longueurs d'interconnexion seront minimisées pour éviter les réflexions en bout de piste. Les pistes ne sont adaptées que pour des liaisons de grande dimension comme on peut en trouver dans la réalisation des bus fond de panier. Les longueurs sont alors de quelques dizaines de centimètres.

## 2 Logique ECL (*Emitter Coupled Logic*)

Pour réaliser des circuits rapides, il est nécessaire d'éviter la saturation des transistors.

Les constructeurs ont donc imaginé des circuits à base de transistors bipolaires fonctionnant en mode non saturé. Cette logique a cependant l'inconvénient majeur de créer une consommation statique très élevée. Elle est de ce fait réservée aux applications nécessitant une fréquence de fonctionnement très importante, au-delà de 200 MHz pour fixer les idées. Le niveau logique bas est de – 1,6 V, le niveau haut est de – 0,8 V. Le fonctionnement détaillé de la porte ECL est décrit ci-après.

Le schéma 3.22 représente une porte NOR dans cette technologie. L'étage d'entrée est formé par un étage différentiel à plusieurs entrées. La base de l'un des bras du différentiel est fixée à un potentiel de référence de – 1,3 V. Si les deux entrées sont à l'état bas, soit de l'ordre de – 1,6 V ou moins, alors le transistor *T3* est conducteur et les transistors *T1* et *T2* sont bloqués. La tension sur le collecteur des transistors *T1* et *T2* est voisine de 0 V. Elle est transmise en sortie à 800 mV de décalage près, par le transistor de sortie *T4*, monté en collecteur commun. La tension de sortie S

est donc à – 0,8 V soit dans l'état logique haut. L'autre sortie reliée à *T3* est alors à un potentiel de – 1,6 V fixé par les valeurs des résistances internes du montage. Quand l'une des entrées est à l'état logique haut, soit – 0,8 V, le transistor correspondant est conducteur et la tension négative présente sur son collecteur est transmise en sortie par le transistor *T4*. Elle a dans ce cas la valeur – 1,6 V, correspondant au niveau bas. L'autre sortie S a alors la valeur – 0,8 V, ce qui correspond à l'état haut. La fonction NOR est donc réalisée. Il faut également remarquer que le montage fournit les deux états logiques complémentaires et que l'alimentation électrique, contrairement à la famille TTL, est négative.

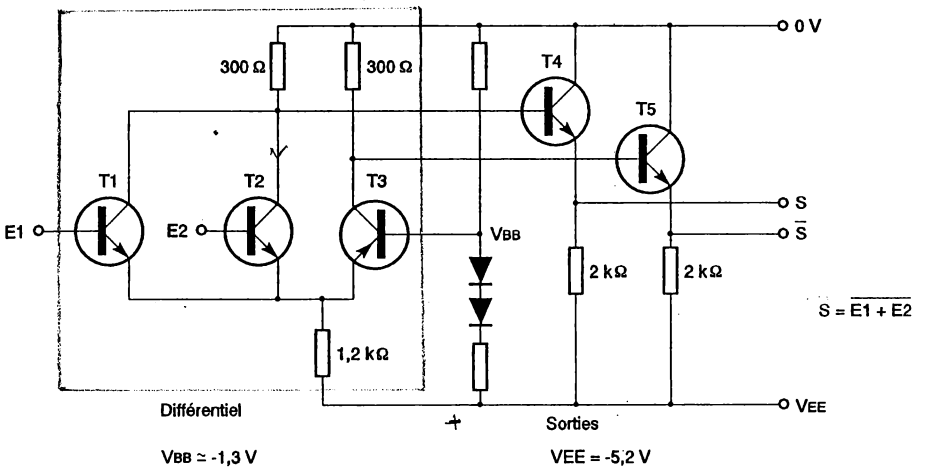



Fig. 3.22
Fonction NOR en technologie ECL.

La famille ECL a donné naissance à diverses sous-familles appelées 10K, 100K et autres, comme l'indique le tableau 3.23 ci-après, qui reprend les mêmes critères que pour la famille TTL. On trouve, chez les constructeurs, des boîtiers intégrant les différentes fonctions logiques étudiées précédemment. Dans la conception de circuits intégrés, on trouve des fonctions de base intégrées sur le silicium qu'il est possible d'interconnecter. Comme pour la technologie TTL, il est possible de relier une sortie à plusieurs entrées.

Ce nombre est cependant fortement réduit quand la fréquence de fonctionnement augmente, et dans la pratique, il faut éviter de connecter plus de cinq entrées.

| | Symbole | Temps de propagation par porte en ns | Consommation statique par porte en mW |
|---|---|---|---|
| ECL 10 k | | 3 | 20 |
| ECL 100 k | | 1 | 50 |
| ECL 100 kH | | 0,5 | 20 |

Fig. 3.23
Familles ECL.

Une particularité de la technologie ECL est sa rapidité. Quand la vitesse de fonctionnement est élevée, ce qui est généralement le cas, les longueurs de connexion supérieures au centimètre peuvent poser des difficultés dues à la désadaptation en bout de ligne. Le montage représenté figure 3.24 permet de s'affranchir de ces difficultés en adaptant la charge en bout de ligne à l'impédance caractéristique présentée.

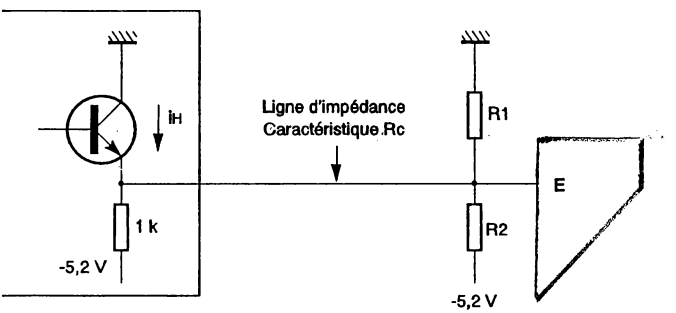



Fig. 3.24
Liaison en technologie ECL.

Il est assez facile de réaliser des pistes de connexion sur circuit imprimé présentant une impédance caractéristique de 100 Ω environ. La charge représentée par les résistances $R_1$ et $R_2$ doit donc être égale à cette impédance caractéristique. On suppose que l'impédance d'entrée du circuit logique a une valeur élevée, ce qui est le cas pour le montage différentiel d'entrée. Si on appelle $i_H$ et $i_B$ les courants débités par le transistor de sortie dans les deux états, on obtient les relations suivantes :

$$\frac{R_1 R_2}{R_1 + R_2} = R_C$$

$$i_H = \frac{5{,}2 - 1{,}6}{R_2} + \frac{5{,}2 - 1{,}6}{1000} + \frac{1{,}6}{R_1}$$

$$i_B = \frac{5{,}2 - 0{,}8}{R_2} + \frac{5{,}2 - 0{,}8}{1000} + \frac{0{,}8}{R_1}$$

La donnée de ces trois équations permet de choisir les valeurs des résistances $R_1$ et $R_2$.

En résumé, il est possible, comme nous l'avons fait pour la logique TTL, de donner les règles de conception en technologie ECL :
– les circuits se connectent de sortie à entrée en ajoutant des résistances de charge sur chaque sortie. C'est la différence principale avec la famille TTL. Le fan-out est rarement supérieur à 5 ;
– l'alimentation des circuits est négative, égale à – 5,2 V ou parfois – 2 V. Elle doit être soigneusement découplée ;
– quand les longueurs de connexion sont élevées, il est nécessaire d'adapter la ligne en ajoutant des résistances de charge. Leurs valeurs sont déterminées par les considérations exposées précédemment ;
– dans tous les cas, la consommation des circuits est élevée et il convient de la calculer pour dimensionner les alimentations basse tension nécessaires ;
– les entrées non utilisées sont définies dans un état logique donné.

## 3 Logique CMOS et BICMOS (*Complementary Metal Oxyde Semiconductor*)

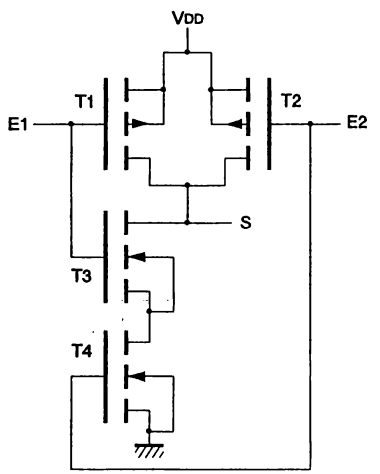
Nous en arrivons à la famille la plus communément utilisée en électronique, la famille **CMOS**. Les autres familles à base de transistors MOS, la logique NMOS et la logique PMOS, sont actuellement peu utilisées. Les avantages de la logique CMOS sont sa faible consommation et sa simplicité comme il a été exposé dans le chapitre consacré aux éléments actifs. Les circuits fonctionnent avec une alimentation positive comprise entre 3 V et 15 V. La valeur de 5 V est très pratiquée. Si on appelle $V_{DD}$ la valeur de la tension d'alimentation, les niveaux logiques haut et bas sont alors donnés par les inégalités :

$$V_{EH} \geq 0{,}7\ V_{DD}$$
$$V_{EB} \leq 0{,}3\ V_{DD}$$

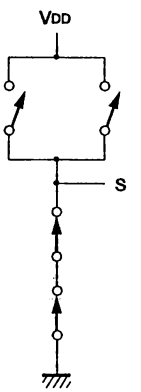
Les marges de bruit sont donc confortables puisque les tensions délivrées par les portes logiques sont proches de 0 V pour l'état bas et proches de la tension d'alimentation pour l'état haut. La famille CMOS offre alors la meilleure immunité aux perturbations de toutes les familles logiques. Si on associe à cela l'avantage de la faible consommation, il est facile de comprendre pourquoi 80 % des circuits actuels sont fabriqués dans cette technologie.

Sans entrer dans des détails de fabrication qui ne sont pas utiles à la conception des systèmes telle que nous la présentons dans cet ouvrage, il est cependant important de comprendre le fonctionnement de la porte NAND de la technologie CMOS. Le schéma est représenté figure 3.25.a : il associe quatre transistors MOS, deux à canal *p*, *T1* et *T2* et deux à canal *n*, *T3* et *T4*.

Comme pour l'inverseur décrit dans le chapitre consacré aux éléments actifs, le fonctionnement de la porte schématisée figure 3.25.b par des interrupteurs est tel qu'aucun courant continu ne puisse être débité dans le montage. La consommation statique est aussi faible que 10 nW par porte. Quand les deux entrées *E1* et *E2* sont à l'état haut, les transistors *T1* et *T2* sont non conducteurs (interrupteurs ouverts) alors que les transistors *T3* et *T4* sont conducteurs (interrupteurs fermés). La sortie *S* est alors dans l'état bas. Si l'une des entrées est dans un état bas, l'un des transistors, *T1* ou *T2*, est conducteur. La tension de sortie est alors à l'état haut puisque reliée à $V_{DD}$ .




*3.25.a*




| E1 | E2 | S |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

*3.25.b*

Fig. 3.25
Porte NAND en technologie CMOS.

Pour terminer cette brève description de la technologie CMOS, il faut rappeler que la consommation est principalement dynamique et donc liée à la capacité d'entrée des portes. Quand la porte est un circuit, la capacité d'entrée est de quelques pF, ne serait-ce que par la présence des liaisons entre la puce et les contacts du boîtier. Quand la porte est un ensemble de transistors sur la puce de silicium, la capacité est celle du transistor. Elle peut être aussi faible que le centième de pF. Il est donc facile de comprendre l'intérêt de l'intégration électronique. La sortance d'une porte en technologie ECL est élevée et si on fait abstraction des limites en fréquence dues à une augmentation de la capacité de charge. Elle peut atteindre des valeurs aussi élevées que 20. Quand les sorties sont de type drain ouvert, la fonction *ET* câblée peut se réaliser.

Pour résumer l'ensemble des propriétés de la famille CMOS, il est possible de donner les propriétés principales des sous-familles produites par les constructeurs.

La même terminologie est utilisée que pour les sous-familles TTL, la lettre C est ajoutée pour indiquer que la technologie est bien de type CMOS. La lettre T est utilisée quand les sorties sont compatibles avec la logique TTL sans avoir à utiliser de circuits de translation. Les circuits CMOS peuvent remplacer les circuits TTL dans le but de réduire la consommation électrique. Cette classification permet d'identifier facilement les circuits équivalents dans les deux technologies. Le tableau 3.26 classe les sous-familles en fonction des deux critères, le temps de propagation et la consommation statique par porte.

| | Symbole | Temps de propagation par porte en ns | Puissance statique dissipée par porte en nW |
|---|---|---|---|
| Standard | 74CXX | 50 | 1 |
| Rapide | 74HCXX | 10 | 0,5 |
| Compatible TTL | 74HCTXX | 10 | 0,5 |
| Avancée | 74ACXX | 3 | 0,5 |
| Avancée et compatible TTL | 74ACTXX | 3 | 0,5 |

Fig. 3.26
Les familles CMOS.

Quelques recommandations générales sont maintenant données pour la conception des systèmes dans la technologie CMOS :
– la sortance d'un circuit peut aller jusqu'à 20 quand les considérations de vitesse ne sont pas primordiales ;
– **il est impératif de définir les états logiques des entrées non utilisées**. Sinon des erreurs peuvent apparaître et des circuits peuvent être détruits. Selon le cas, elles seront reliées à la masse ou au 5 V ;
– les découplages d'alimentation sont moins critiques que dans les technologies TTL et ECL. Un découplage de 100 nF pour un ensemble de circuits peut être suffisant. Le découplage basse fréquence de l'ensemble de la carte reste indispensable ;
– la technologie des plans de masse peut également être simplifiée, et dans les applications les plus simples, quand il s'agit de réduire le coût au maximum, une simple piste assez large peut faire office de plan de référence. Quand les interconnexions sont nombreuses, la technologie multicouches reste intéressante et l'une des couches doit être réservée à la masse ;
– la consommation statique est négligeable et la seule consommation à prendre en compte est la consommation dynamique. Elle est de l'ordre de 1 mW par porte à 10 MHz ;
– les circuits MOS présentent cependant l'inconvénient d'être sensibles à l'électricité statique. Des diodes de protection sont

maintenant placées en entrée des circuits et évitent les destructions par surtension mais les composants MOS restent fragiles et il est conseillé de les manipuler avec précaution.

La famille BICMOS est une version plus récente de la logique CMOS. Elle associe dans une même technologie des transistors MOS et des transistors bipolaires et permet ainsi d'augmenter la vitesse de fonctionnement sans pour autant augmenter la consommation. De plus, il est possible dans cette technologie de réaliser simultanément des fonctions logiques et analogiques.

Pour terminer cet important paragraphe, il est possible de résumer dans le tableau 3.27 les principales propriétés des familles logiques TTL, ECL et CMOS. Les chiffres donnés dans ce tableau sont des valeurs moyennes et relatives à la technologie de 1996.

| | TTL | ECL | CMOS |
|---|---|---|---|
| Alimentation | 5 V | -5,2 V | 3 à 15 V |
| Sortance | 10 | 5 | 20 * |
| Vitesse de fonctionnement maximale | 50 MHz | 1 GHz | 200 MHz |
| Consommation statique par porte | 10 mW | 50 mW | 0 |
| Consommation dynamique à 1 MHz | | | 1 mW |
| Immunité | moyenne | faible | élevée |

* (moins si la fréquence est élevée)

Fig. 3.27
Les familles logiques.

## Les mémoires

La fonction **mémoire** est une des plus importantes et avant même de décrire les différents types de circuits, il est utile de préciser le rôle des mémoires dans les systèmes. La plupart des systèmes électroniques actuels comportent des circuits de type microprocesseur. Ces circuits sont étudiés dans un chapitre spécial. A ce stade de la description des systèmes, il suffit de savoir que la fonction des ces circuits complexes est de dérouler une suite d'opérations dans le temps, faisant intervenir éventuellement des données extérieures au système électronique considéré. Dans une régulation de chauffage contrôlée par microprocesseur, les données extérieures seront par exemple les températures mesurées par des sondes. D'autres données sont internes mais peuvent être modifiées, par exemple la température de consigne. Le programme consiste à comparer de manière continue températures mesurées et température de consigne puis, après un calcul mathématique, à commander la combustion de la chau-

dière et le mélange entre l'eau chaude et l'eau froide dans l'installation. Cet exemple montre que deux types de données sont mis en mémoire dans le système électronique, d'une part la séquence d'instructions permettant de commander convenablement la régulation et d'autre part les données ou les valeurs de consigne.

La séquence d'instructions sera toujours la même mais ne devra pas être perdue, par exemple à la suite d'une coupure de puissance électrique. On dit alors que les instructions sont mises en mémoire dans un mémoire de type programme. Les données en provenance des sondes de températures et les consignes peuvent être perdues puisqu'elles sont remises à jour en permanence. Elles sont mises en mémoire dans une mémoire de type donnée.

Les deux types de mémoires sont donc :

**Mémoire programme** ou **ROM (*Read Only Memory*) :**
- lecture seule,
- ne doit pas être modifiée suite à une coupure d'alimentation.

**Mémoire de données** ou **RAM (*Random Access Memory*) :**
- lecture et écriture,
- rapidité d'accès nécessaire,
- les données peuvent être perdues suite à une coupure d'alimentation.

## 1 Mémoire et adressage

Une mémoire se présente donc comme un ensemble d'éléments de stockage de données binaires (bit ou groupe de bits). Les données doivent être accessibles soit pour être changées (écriture), soit pour être utilisées (lecture). De manière plus imagée, elle est équivalente à un immeuble dont chaque appartement a une adresse et dont le contenu est la donnée binaire considérée. La donnée est un bit, ou plus communément un groupe de 8 bits, appelé **octet** ou **byte**. Les adresses sont codées en binaire et le nombre total d'emplacements ou d'adresses est la taille de la mémoire. Elle est exprimée en puissance de deux selon les données suivantes :

$2^{10}$ : 1 024 = 1 kilo octet
$2^{12}$ : 4 096 = 4 kilo octets
$2^{20}$ : 1 048 576 = 1 Méga octet

Les terminologies *kilo* et *méga* permettent de s'approcher des valeurs décimales mais correspondent à des puissances de deux. Selon que le contenu de l'emplacement est un bit ou un byte, on parlera par exemple d'une mémoire de 1 Mbit ou de 1 Mbyte. La taille des mémoires électroniques ne cesse d'augmenter et des capacités de 16 Mbits sont disponibles.

La figure 3.28 représente les entrées et les sorties d'un circuit mémoire. En général, les mêmes broches sont utilisées pour les données entrantes donc écrites et les données sortantes donc

lues. Un certain nombre de lignes d'adresses permettent de choisir l'emplacement mémoire. Des signaux de contrôle permettent de choisir le mode de fonctionnement : mode lecture ou mode écriture.

Ces signaux sont en général actifs à l'état bas et sont notés $\overline{WE}$ (*Write Enable*) pour l'écriture et $\overline{OE}$ (*Output Enable*) pour la lecture. Le signal $\overline{CS}$ (*Chip Select*) permet de rendre le circuit opérationnel ou non. Son utilisation sera décrite ultérieurement.

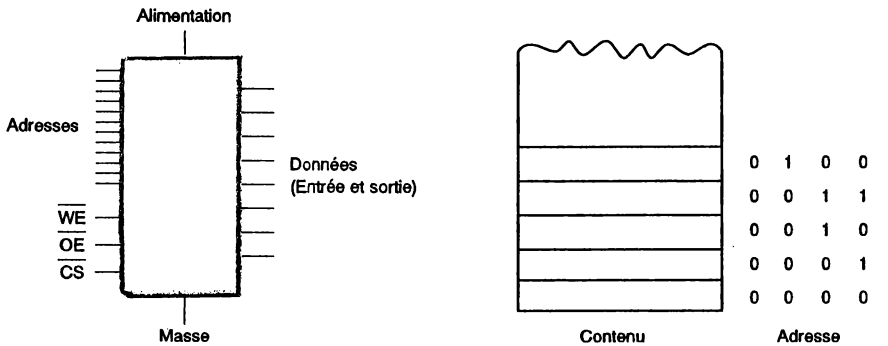
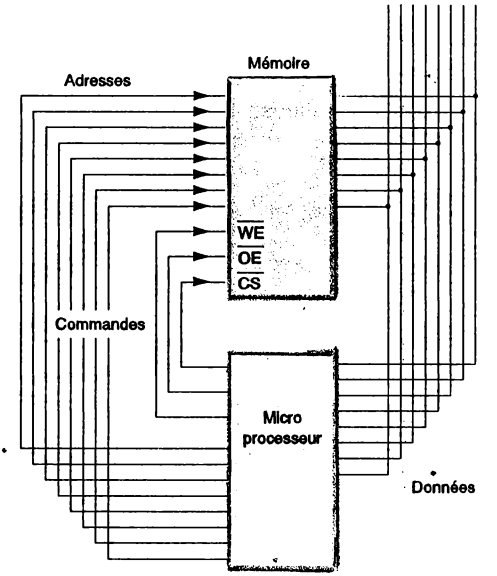



Fig. 3.28
Mémoire type.

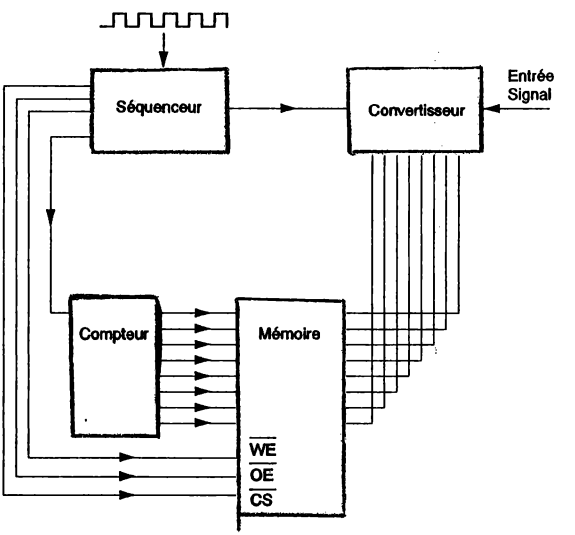
La figure 3.29 représente deux utilisations classiques des mémoires.

Dans la figure 3.29.a, les données en provenance d'une source extérieure (capteur, clavier, microprocesseur, etc.) sont appliquées en entrée de la mémoire via un bus de données. En parallèle, l'adresse de stockage est choisie par un circuit intelligent externe de type microprocesseur. Les données pourront ensuite être lues sur le même bus par le microprocesseur du système.

Dans ce mode de fonctionnement, l'adresse est choisie librement et on parle d'**accès aléatoire**. Dans le schéma 3.29.b, les données ne sont plus mises en mémoire librement mais dans l'ordre des adresses croissantes. Cette architecture est adaptée par exemple à la mise en mémoire d'une image numérisée ou d'un signal échantillonné et numérisé. Dans ce cas, les adresses défilent et sont créées en sortie d'un compteur de taille adaptée. Un circuit logique commandé par une horloge permet de synchroniser les données, les adresses et les signaux de commande. De la même manière que dans le cas précédent, les données pourront être relues mais séquentiellement. On parle alors d'**accès séquentiel**. Les signaux de commande ne doivent être émis que lorsque les adresses et les données sont bien établies. Les constructeurs définissent donc des temps de set-up pour le bon fonctionnement de la mémoire.




*3.29.a*




*3.29.b*

Fig. 3.29
Utilisations d'un circuit mémoire.

La figure 3.30 visualise ces temps dans une opération d'écriture. Les chronogrammes représentent les deux états binaires possibles des bits d'adresses et de données.

Dans la pratique, il est parfois nécessaire d'utiliser plusieurs circuits mémoire car le champ d'adressage est trop important (figure 3.31).

L'architecture de la figure 3.31.a montre comment il est possible d'associer plusieurs mémoires pour créer une mémoire de taille plus importante. Le rôle du Chip Select est précisé à cette occasion. Dans une opération de lecture, la transition descendante du Chip Select fait apparaître les données correspondant à l'adresse sur les lignes de sortie. Dans une opération d'écriture, les signaux *CS* et *WE* sont positionnés à chaque changement d'adresse. La donnée présente est alors mise en mémoire. Nous disposons de mémoires de 16 kbytes et il est nécessaire de réaliser une mémoire de 64 kbytes. Il faut donc associer quatre mémoires. Le Chip Select permet de sélectionner ou non la mémoire considérée. Il suffit donc de décoder les 16 lignes d'adresses correspondant aux 64 k d'espace mémoire pour en extraire les 14 lignes d'adressage des mémoires 16 k et les 4 commandes de Chip Select. L'allocation mémoire est représentée figure 3.31.b.

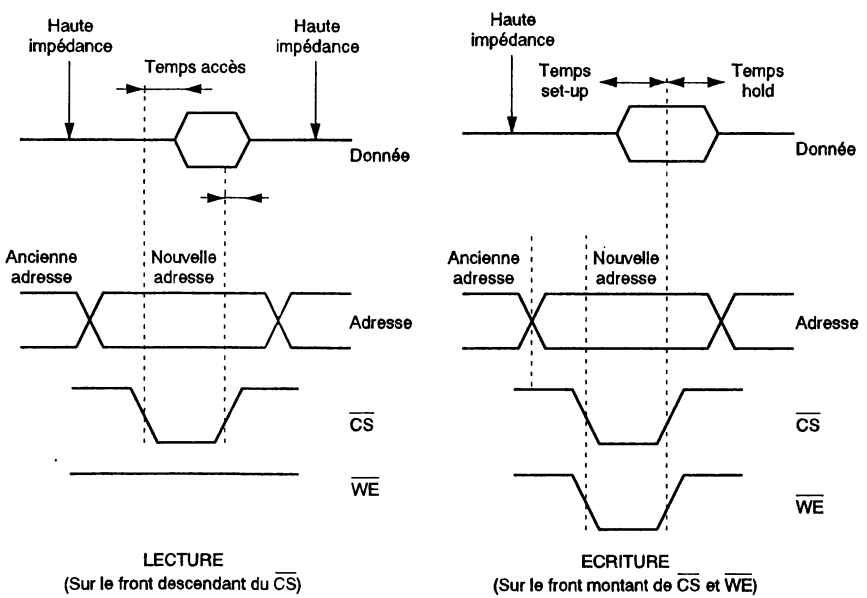



Fig. 3.30
Lecture et écriture dans une mémoire.

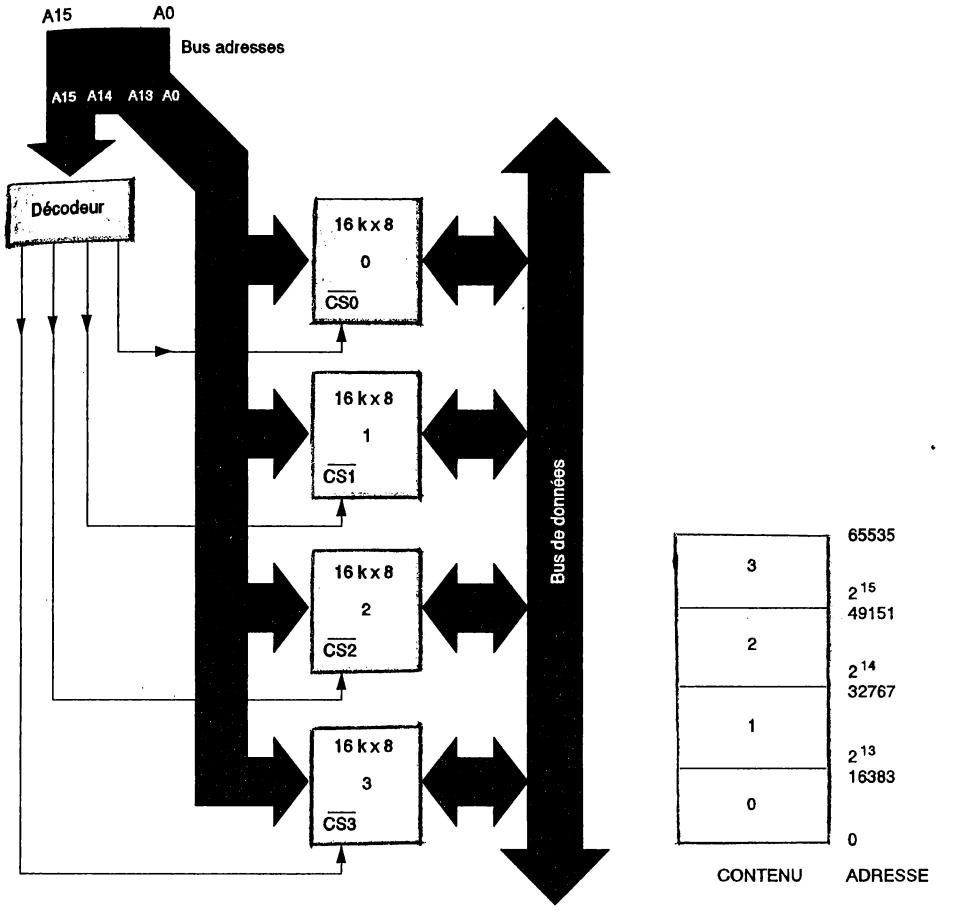



Fig. 3.31
Association de mémoires.

## 2 Random Access Memory (RAM)

Ce type de mémoire permet d'écrire et de lire des données utiles à un système électronique. Le temps d'accès, c'est-à-dire le temps séparant l'ordre de lecture ou d'écriture de la mise à disposition de données stables sur les lignes de sortie, doit être le plus court possible et contribue largement à la performance du système. Des temps d'accès de 10 ns sont maintenant courants en technologie CMOS.

Les mémoires RAM sont de deux types : les mémoires **statiques** et les mémoires **dynamiques.**

Les mémoires statiques sont formées à partir de bistables électroniques et il suffit d'alimenter le circuit pour garder l'information. Il faut environ six transistors MOS pour former une cellule.

Les mémoires dynamiques sont formées à partir de cellules plus simples puisque l'information est conservée comme la tension de charge d'un condensateur. Il suffit alors de trois transistors pour former une cellule. La contrepartie de cette simplicité est dans la nécessité de rafraîchir périodiquement la mémoire. Le condensateur se décharge en effet au bout d'un certain temps et si rien n'était fait, il perdrait toute l'information mémorisée. Une logique externe, pour les circuits de conception ancienne, réalise ces opérations périodiques de rafraîchissement et la mémoire est indisponible pendant environ 5 % du temps.

La circuiterie annexe est plus complexe et les temps d'accès sont plus longs que pour les mémoires statiques. La taille de la mémoire peut cependant être considérable et des valeurs aussi élevées que 64 Mbits sont disponibles dans un seul circuit. L'écriture des données dans les mémoires RAM se réalise en général à partir de l'application elle-même via un programme d'écriture. Cela est évident dans le cas d'une carte de numérisation d'image. Les données sont écrites à partir du signal vidéo. Le tableau 3.32 donne les caractéristiques de quelques produits du commerce.

| | Taille | Temps d'accès | Rafraîchissement |
|---|---|---|---|
| Statique (SRAM) | 16 Mbit max. | 10 ns min. | non |
| Dynamique (DRAM) | 64 Mbit max. | 50 ns min. | oui |

Fig. 3.32
Mémoires RAM.

## 3 Read Only Memory (ROM), EPROM et EEPROM

Comme leur nom l'indique, ces mémoires peuvent uniquement être lues. Elles sont écrites à la fabrication ou dans une phase de préparation spéciale appelée phase de programmation. Cette particularité les réserve au stockage des programmes ou des tables de conversion fixes (tables de fonctions mathématiques ou tables de génération graphique de caractères).

Elles sont de deux types : les mémoires programmées par masque et les mémoires programmables par fusibles. Les mémoires ROM programmées par masque sont programmées à la fabrication même du circuit intégré et les connexions électriques nécessaires

sont établies dans le procédé de fabrication. Cette technique doit être réservée à la production de grandes séries pour amortir le coût de réalisation des masques lithographiques. La figure 3.33 montre une réalisation possible à partir de transistors MOS.

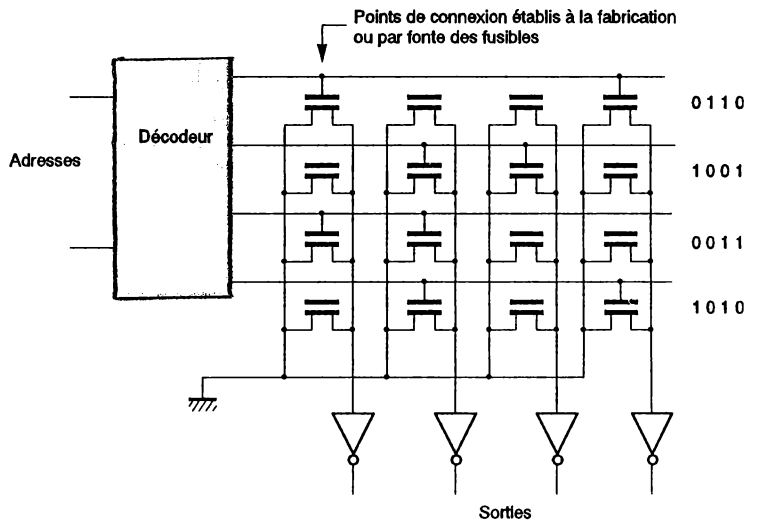



Fig. 3.33
Structure d'une mémoire ROM.

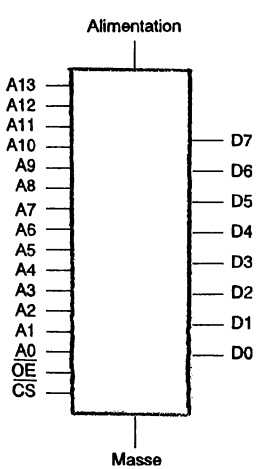



Fig. 3.34
Mémoire 32 kilobytes.

Quand l'adresse est présente, le circuit de décodage polarise la ligne électrique correspondante. Les transistors sont conducteurs et les sorties reliées électriquement sont au potentiel bas. Le schéma 3.34 montre le brochage d'une mémoire 32 kbytes. On reconnaît les bits d'adresses, les 8 bits de sortie généralement dans une technologie trois états et les deux signaux de contrôle, $\overline{OE}$ et $\overline{CS}$, l'un pour autoriser la sortie de la donnée et l'autre pour autoriser l'utilisation du circuit. Quand ce niveau logique est à l'état haut, le circuit est en veille et sa consommation est réduite. Il est donc possible d'éteindre les circuits non utilisés.

La même technique de programmation peut être mise en œuvre avec des fusibles électriques réalisés sur le circuit intégré. La programmation s'effectue alors en grillant ces fusibles. Pour cela, on applique au dispositif une tension définie pour faire fondre le fusible concerné. Il est possible d'effectuer cette opération manuellement mais les fabricants ont développé des programmateurs qui permettent, à partir d'un clavier, d'écrire dans la mémoire les données voulues et de contrôler que l'opération s'est déroulée correctement.

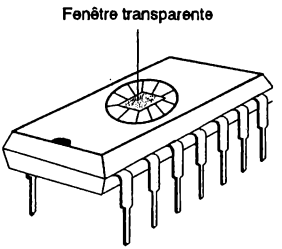



Fig. 3.35
EPROM.

Toutes ces opérations étant irréversibles, les constructeurs ont imaginé des dispositifs ROM particuliers donnant à l'utilisateur plus de souplesse. Ces dispositifs sont les EPROM ou EEPROM. Le procédé d'écriture est le même que précédemment mais il est possible d'effacer les données en exposant le circuit pendant une dizaine de minutes à la lumière des rayons ultraviolets. Le circuit est alors utilisable pour une nouvelle opération d'écriture. La figure 3.35 représente un tel circuit. On y remarque une fenêtre transparente aux ultraviolets laissant voir la puce de silicium. La figure 3.36 représente les signaux électriques aux bornes pendant les opérations de programmation ou de lecture. La tension de programmation est de l'ordre de 21 V. Ce type de mémoire présente deux inconvénients : il est nécessaire dans une opération de correction de sortir le circuit mémoire du système pour le reprogrammer. Toutes les données sont effacées aux ultraviolets et quand bien même, quelques données sont à modifier, il faut reprogrammer complètement la mémoire.

Les constructeurs ont donc imaginé une technologie plus évoluée appelée EEPROM, *Electricaly Erasable* PROM. Une simple tension de valeur suffisante permet d'effacer la mémoire.

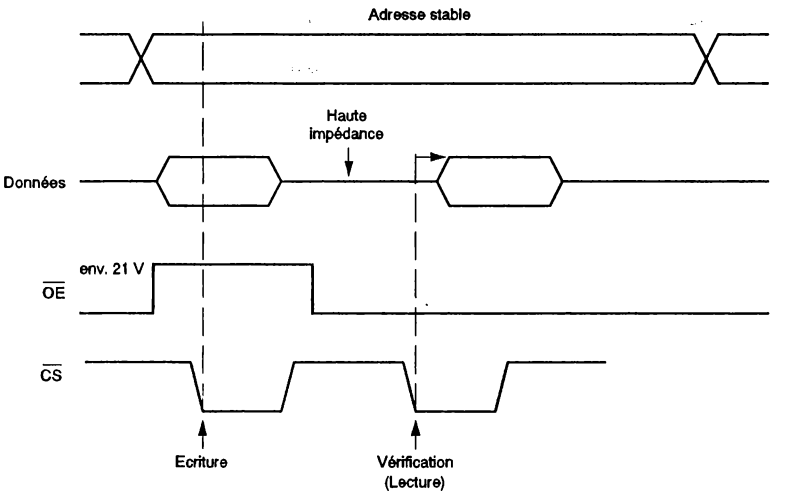



Fig. 3.36
Programmation d'une EPROM.

## 4 Mémoire flash

Apparue plus récemment, c'est en fait la version la plus moderne des EEPROM. Le terme *flash* vient du fait que l'ensemble de la zone mémoire peut être effacé et non pas de façon sélective comme dans le EEPROM. Elle offre les avantages des EEPROM :

elle est non volatile, c'est-à-dire qu'elle n'est pas effacée en cas de coupure d'alimentation. Elle est reprogrammable électriquement. Elle est disponible à haute densité, 16 et 32 Mbits. Son utilisation principale est de replacer les EPROM afin de pouvoir mettre facilement à jour les programmes. Les temps d'accès restent cependant relativement élevés, de l'ordre de 100 ns. Le tableau 3.37 présente quelques produits industriels.

Fig. 3.37
Mémoires flash.

| | Capacité en bits | Temps d'accès en ns | Temps de reprogrammation en ms |
|---|---|---|---|
| INTEL 28FO325A | 32 M | 100 | 3 / Octet |
| SHARP LH28FO16SUT | 16 M | 70 | 700 pour 64 kbytes |
| SGS THOMSON M28V411 | 4 M | 150 | |

## 5 Mémoires magnétiques et optiques

Nous n'avons jusqu'ici considéré que les mémoires de type électronique. Leur capacité de stockage importante (64 Mbits pour les plus développées) n'est cependant pas suffisante pour mettre en mémoire les gigantesques programmes utilisés actuellement en informatique. Un logiciel de traitement de texte a, par exemple, une taille de 10 Mbytes. Il est donc nécessaire de disposer de mémoires de plus fortes capacités. Ces mémoires sont de deux types : les mémoires magnétiques et les mémoires optiques.

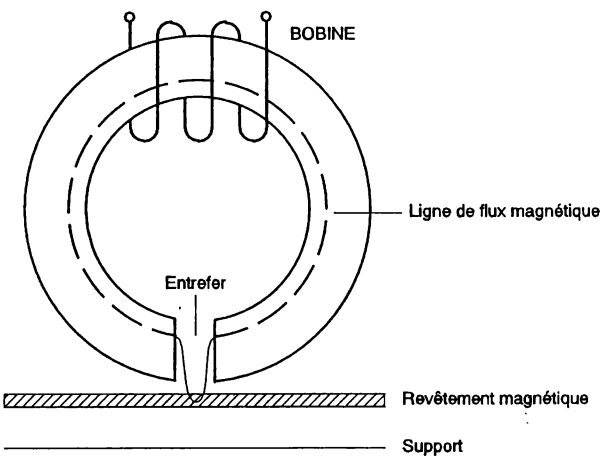



Fig. 3.38
Mémoire magnétique.

Le principe de fonctionnement d'une mémoire magnétique est représenté figure 3.38. Le signal électrique correspondant au bit à mettre en mémoire est appliqué au transformateur. Les lignes de flux magnétique pénètrent en partie dans le revêtement magnétorésistif déposé sur le disque support et des points d'aimantation sont créés.

À l'inverse quand il s'agit de lire le bit mis en mémoire, il suffit de mesurer la tension électrique induite sur le transformateur quand le point aimanté traverse l'entrefer.

La figure 3.39 montre que l'association d'un mouvement de rotation du disque et d'un mouvement de translation de la tête permet d'adresser toute la surface du disque.

Quand le disque est fixe et protégé dans un compartiment étanche à la poussière, il est appelé **disque dur**. Les capacités sont alors comprises entre 20 et 10 Gbytes. Les temps d'accès sont de l'ordre de 20 ms. A ce sujet, il faut noter que le mode d'accès est de type séquentiel. La tête de lecture ayant été positionnée, il est ensuite aisé d'écrire ou de lire les emplacements correspondant au mouvement de rotation du disque. Le temps d'accès est donné pour un positionnement complet de la tête de lecture ou d'écriture.

Quand le disque peut s'enlever de son support, il se présente alors sous la forme d'un **disque souple**. La capacité est alors limitée à 1.44 Mbyte.

Les ordinateurs personnels comportent généralement un lecteur de disque dur et un lecteur de disque souple. Le disque dur sert à conserver les programmes d'usage général et les disques souples servent à conserver les programmes d'usage particulier ou les données.

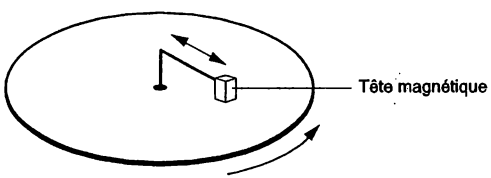



Fig. 3.39
Stockage sur disque.

Les mémoires optiques sont une évolution récente des mémoires magnétiques à haute densité. L'écriture des données se fait alors en chauffant à l'aide d'un microlaser une zone du support, comme le représente la figure 3.40. Il est alors possible de modifier l'aimantation locale. La lecture utilise un principe physique particulier, l'effet Kerr. La lumière d'un laser, réfléchie par la zone du

disque, est analysée par un photodétecteur. La mesure de la polarisation varie avec l'aimantation de la zone. Comme dans l'enregistrement magnétique, la rotation du disque et la translation de la tête permettent de décrire toute la surface du disque.

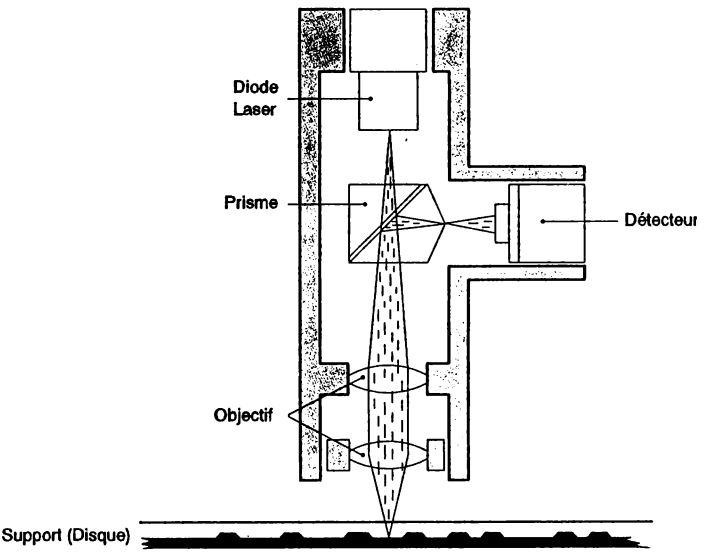



Fig. 3.40
Mémoire optique.

## ENTRAÎNEMENT

**1** Qu'est-ce que découpler un circuit ? Comparez les familles TTL, ECL et CMOS.

**2** Deux circuits ont les caractéristiques suivantes :

| circuit A | | circuit B | |
|---|---|---|---|
| $V_{SH}$ | = 3 V | $V_{SH}$ | = 3 V |
| $V_{EH}$ | = 2,8 V | $V_{EH}$ | = 2,5 V |
| $V_{SB}$ | = 0,5 V | $V_{SB}$ | = 0,7 V |
| $V_{EB}$ | = 0,9 V | $V_{EB}$ | = 0,9 V |
| $t_P$ | = 5 ns | $t_P$ | = 7 ns |

Quel circuit présente la meilleure immunité au bruit ? Quel circuit peut fonctionner à la vitesse la plus élevée ?

**3** Un circuit CMOS a une impédance de sortie de 1 kΩ. Une porte a une capacité d'entrée de 5 pF. Si la fréquence de fonctionnement est de 10 MHz, calculez la sortance.

**4** Démontrez que la puissance dissipée par la commutation d'une porte à la fréquence *f* est :

$$P = f \cdot C \cdot V^2$$

*V* est la différence de potentiel entre les deux états logiques et *C* est la capacité d'entrée.

**5** Comparez les logiques TTL, CMOS et ECL.

**6** Dans quel cas une logique TTL peut-elle consommer moins qu'une CMOS ?

**7** Comment traiter les entrées non utilisées dans les technologies TTL, CMOS et ECL ?

**8** Dans une application à base de microprocesseur, il faut traiter une image de 1 024 fois 1 024 points. Chaque niveau de gris est codé sur 8 bits. Quelle est la taille de la mémoire RAM à utiliser ?

**9** Expliquez la différence entre un disque dur et un disque souple.

**10** Expliquez les termes RAM, ROM, mémoire statique et mémoire dynamique.

**11** Comparez ROM et circuits programmables.

Chapitre 4

# MICROCONTRÔLEURS, MICROPROCESSEURS ET MICRO-ORDINATEURS

**SOMMAIRE**


- Définition des termes
- Exemple d'un microcontrôleur : description d'un 8051
- Exemple d'un microprocesseur : description de la famille 8086
- Le micro-ordinateur : architecture, caractéristiques, utilisation


*Plusieurs ouvrages seraient nécessaires à l'étude complète des trois sous-ensembles abordés dans ce chapitre : microcontrôleur, microprocesseur et micro-ordinateur. En conséquence, ce chapitre constitue uniquement un condensé qui permettra au futur concepteur de choisir, parmi les trois solutions (microcontrôleur, microprocesseur ou micro-ordinateur), le meilleur compromis pour son application.*
*La première partie de ce chapitre sera consacrée à l'énumération d'un certain nombre de termes de vocabulaire, nécessaires à la description de ces éléments.*
*Les exemples montreront clairement les domaines d'application et la séquence de mise en œuvre d'un processeur.*

## Définition des termes

### 1 Largeur du mot

À l'intérieur d'un microprocesseur ou microcontrôleur, les informations sont stockées et manipulées en un groupe de bits. La taille de ce groupe utilisé est appelée **largeur**.

Cette largeur est un des facteurs qui détermine la puissance de calcul puisque cette caractéristique fait référence à la quantité de données que le processeur peut manipuler simultanément à un instant donné.

Les premiers processeurs avaient une largeur de 4 bits. Ils ne sont utilisés aujourd'hui que dans les applications simples nécessitant une faible puissance de calcul.

La plupart des microcontrôleurs ont une largeur de 8 bits, et les plus puissants d'entre eux, 16 bits. La plupart des microprocesseurs actuels ont une largeur de 32 bits. Il existe des processeurs avec des largeurs de 64 ou même 128 bits qui sont réservés à des applications spécialisées.

La largeur détermine le nombre maximum de bits de données qui transite sur le bus de données à un instant. Cette largeur ne limite pas les possibilités et la précision d'un calcul effectué par la machine. Sur un processeur 8 bits, les calculs seront simplement plus longs, car demandant un nombre de manipulations plus élevé, que sur un processeur 16 ou 32 bits.

## 2 CPU (Central Processing Unit)

La CPU est le cœur des microcontrôleurs et microprocesseurs. Elle est responsable de tous les calculs sur les données, dans la largeur du processeur, et de toutes les manipulations sur les données. La CPU contrôle le déroulement du programme, la lecture de l'instruction à exécuter, l'interprétation de l'instruction, l'exécution et le passage à l'instruction suivante.

## 3 Les mémoires

La CPU est liée via un ou plusieurs bus, adresses, données et bus de contrôle par exemple, à deux types de mémoires.

Les mémoires à écriture-lecture sont des mémoires RAM qui contiennent les données ou les résultats intermédiaires. Ces mémoires sont volatiles, c'est-à-dire que leur contenu est affecté par la disparition de l'alimentation.

Les mémoires à lecture seulement sont des mémoires ROM ou EPROM qui contiennent les instructions à exécuter. Ces mémoires sont non volatiles. Leur contenu est insensible aux pertes brèves ou prolongées de la tension d'alimentation.

Ces informations préliminaires sont tout juste suffisantes pour comprendre les distinctions entre microcontrôleur et microprocesseur.

## 4 Le microcontrôleur

Le microcontrôleur est un composant qui regroupe au moins une **CPU**, une certaine quantité de mémoire programme, de mémoire données et des interfaces d'entrées/sorties avec l'environnement extérieur.

Le microcontrôleur peut fonctionner avec seulement trois composants externes et pour cette raison, il est souvent dit monochip (un seul circuit). Ceci ne signifie pas qu'une application à microcontrôleur nécessite un seul circuit mais plutôt un seul circuit au minimum.

Les microcontrôleurs courants ont des largeurs de 4, 8 et 16 bits. Les microcontrôleurs de type 8 bits sont les plus répandus : familles 6800 Motorola ou 8051 Intel par exemple.

Par sa capacité en regard de ses performances, le microcontrôleur est un organe à faible coût.

L'écriture du programme, son test et sa mise au point requièrent un certain nombre d'outils à faible coût, dont le rôle sera détaillé ultérieurement.

L'utilisation de ces outils ne nécessite que des connaissances de base, en conséquence les temps de développement sont réduits.

En contrepartie, les microcontrôleurs ont une faible puissance de calcul par rapport aux microprocesseurs (µP). Cette différence est due en partie à la largeur µC 8 bits contre µP 32 bits par exemple.

Toutes ces caractéristiques ouvrent aux microcontrôleurs un très vaste domaine d'application. Ils sont judicieusement utilisés pour remplacer toutes les circuiteries en logique câblée. Ceci entraîne une réduction des coûts de câblage et de maintenance.

Le microcontrôleur exécute un programme stocké dans sa mémoire, le changement ou la modification du programme permet une évolution du produit dans lequel est intégré le microcontrôleur. Cette évolution est impensable si la circuiterie est réalisée en logique câblée.

Les applications des microcontrôleurs sont extrêmement nombreuses, leur faible coût les dédie plus particulièrement aux applications grand public.

En examinant les appareils que nous utilisons quotidiennement (récepteurs vidéo, TV, magnétoscope, réveil, autoradio, machines à laver, automobile), chacun peut imaginer la tâche confiée au microcontrôleur.

Dans la plupart des cas, le microcontrôleur reçoit les informations en provenance d'un clavier, contrôle un processus et affiche des résultats. En automobile, les microcontrôleurs peuvent être affectés à la gestion de la climatisation. Les microcontrôleurs ne sont pas seulement employés dans les applications grand public simples, ils sont bien présents dans les domaines industriels, militaires et dans les télécommunications.

## 5 Le microprocesseur

Le microprocesseur est un composant qui se résume essentiellement à une CPU. Contrairement au microcontrôleur, il ne comporte aucune mémoire ni circuiterie d'interface d'entrées/sorties. Le microprocesseur ne peut donc fonctionner seul. Il doit être connecté aux mémoires programmes, mémoires données et aux circuits d'interface.

Pour les microprocesseurs courants, la largeur est de 32 bits. En conséquence, ces circuits sont particulièrement bien appropriés lorsqu'on a besoin d'une forte puissance de calcul.

Les champs d'application des microcontrôleurs et microprocesseurs peuvent donc se recouvrir, le microprocesseur étant utilisé lorsque les calculs doivent être effectués rapidement ou en temps réel (injection électronique ou ABS en automobile par exemple).

Cette forte puissance de calcul n'est acquise qu'en sacrifiant le paramètre coût.

Contrairement au microcontrôleur, avec un microprocesseur 32 bits, les temps de développement sont allongés, et l'élaboration des programmes demande des connaissances approfondies.

Ceci implique, en général, une formation particulière pour un type de microprocesseur.

Dans la plupart des cas, les microprocesseurs sont mis en service pour réaliser des calculateurs ou micro-ordinateurs.

L'exemple de micro-ordinateur le plus connu est la machine que l'on a coutume d'appeler PC (Personal Computer).

Mais les calculateurs peuvent aussi être des machines dédiées à une application spécifique.

Le traitement d'image appliqué au domaine médical est un bon exemple d'application des microprocesseurs sur une machine dédiée à cette application.

## 6 Le micro-ordinateur

Le type de micro-ordinateur le plus répandu est le PC dû à l'origine à IBM.

Il s'agit donc de l'application principale des microprocesseurs. Les premiers micro-ordinateurs PC étaient équipés de microprocesseurs Intel 8086/8088.

Les machines ont suivi l'évolution des processeurs 80286-80386-80486 puis Pentium.

Jusqu'au processeur 80386, il était possible d'ajouter un coprocesseur mathématique 8087 ... 80387 qui augmentait les performances de la machine.

À partir du processeur 80486, 486 en abrégé, le coprocesseur mathématique et le processeur sont localisés dans le même boîtier et constituent une seule et unique entité.

Simultanément, tous les composants et sous-ensembles inclus dans un micro-ordinateur ont suivi une évolution technologique accélérée due à la réussite commerciale du produit.

Le micro-ordinateur est une machine d'usage général qui comprend une large mémoire RAM et une mémoire de masse,

disque dur, CD-ROM, dans lesquelles sont stockés les programmes de l'utilisateur.

Cet ouvrage étant dédié à l'électronique, il est intéressant d'énumérer tous les cas les plus importants pour lesquels cet outil simplifie la tâche du concepteur.

Le micro-ordinateur peut être utilisé pour toutes les opérations de *AO (AO signifie assisté par ordinateur) et est relatif à un vaste domaine : conception, simulation, etc.

La première aide apportée par le micro-ordinateur est l'établissement du schéma de principe. Outre la clarté et la propreté du schéma, les logiciels de dessin, ou schématique assistée par ordinateur, permettent souvent un contrôle approfondi des règles électriques : détection des courts-circuits, détection des entrées non connectées, de sorties reliées entre elles par exemple.

Le fichier schéma de principe peut, après une transformation ad-hoc, être soumis à un simulateur logique ou à un simulateur analogique.

Ce même fichier schéma est utilisé pour le tracé des pistes d'un circuit imprimé. Le circuit imprimé est alors rigoureusement conforme au schéma de principe.

Finalement, le micro-ordinateur est utilisé pour le développement des programmes associés aux microcontrôleurs ou microprocesseurs.

## Exemple d'un microcontrôleur : description d'un 8051

Les microcontrôleurs de la famille 8051 ont été choisis car ils sont extrêmement répandus et ont donné naissance à un grand nombre de descendants.

Ces descendants comportent toujours le même cœur de base auquel on ajoute des fonctions complémentaires comme des convertisseurs A/D ou D/A par exemple (voir chapitre 7).

Nombreux sont les fabricants de semi-conducteurs qui peuvent offrir ce microcontrôleur dû à l'origine à Intel.

On trouve les plus grands constructeurs : Siemens, Matra Harris, Philips, OKI, Fujitsu, etc.

Les microcontrôleurs 8051 sont des microcontrôleurs 8 bits dont l'architecture interne est représentée à la figure 4.1.

Le microcontrôleur de base comporte quatre unités :
- l'unité centrale ou CPU pour Central Processing Unit,
- la mémoire de programme ROM,
- la mémoire de données RAM,
- les ports d'entrées/sorties 8 bits.

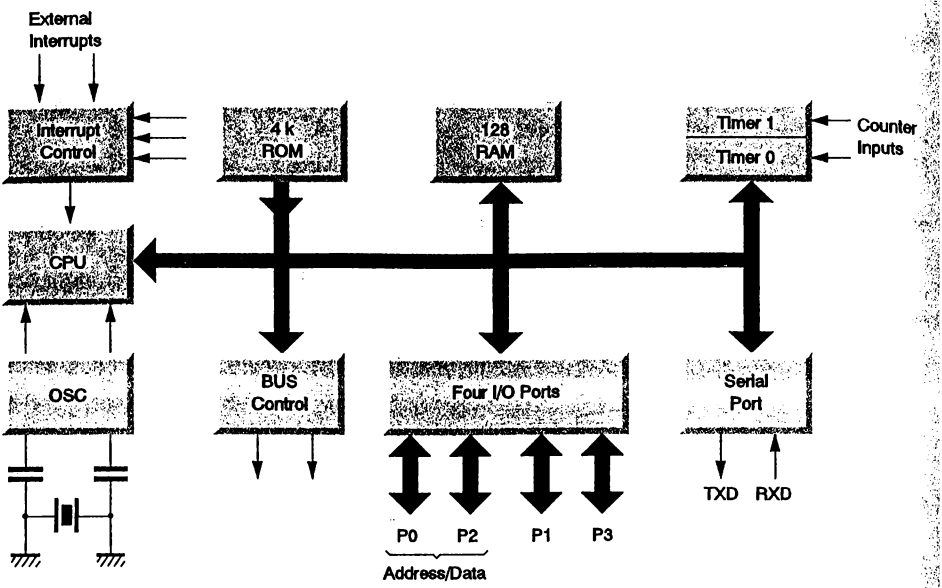



Fig. 4.1
Architecture interne des microcontrôleurs de la famille 8051.

Dans le cas de la figure 4.1, la CPU est cadencée par un oscillateur dont la fréquence est déterminée par un quartz externe. Les mémoires programmes et données sont internes mais il existe d'autres configurations, mémoires externes, qui seront examinées ultérieurement.

Les instructions pour le contrôle des opérations sont stockées dans la mémoire de programme. Les données et les opérandes à traiter sont enregistrées dans la mémoire de données. Chaque instruction lue dans la mémoire de programme est exécutée par l'unité centrale. Les instructions sont lues et exécutées en séquence par l'unité centrale.

Les extensions internes au microcontrôleur peuvent être :
- des compteurs (timer 0 et timer 1),
- des lignes série - **UART** (voir chapitre 10),
- des générateurs de signaux modulés en largeur **PWM**,
- des convertisseurs A/D-D/A,
- des temporisateurs (compteurs et base de temps),
- des circuits d'interface spécifique (bus **I2C** par exemple), etc.

## 1 Généralités sur le microcontrôleur de base

L'architecture détaillée du microcontrôleur de base est donnée à la figure 4.2 où un bus interne relie chaque unité. Les divers composants sont détaillés ci-après.

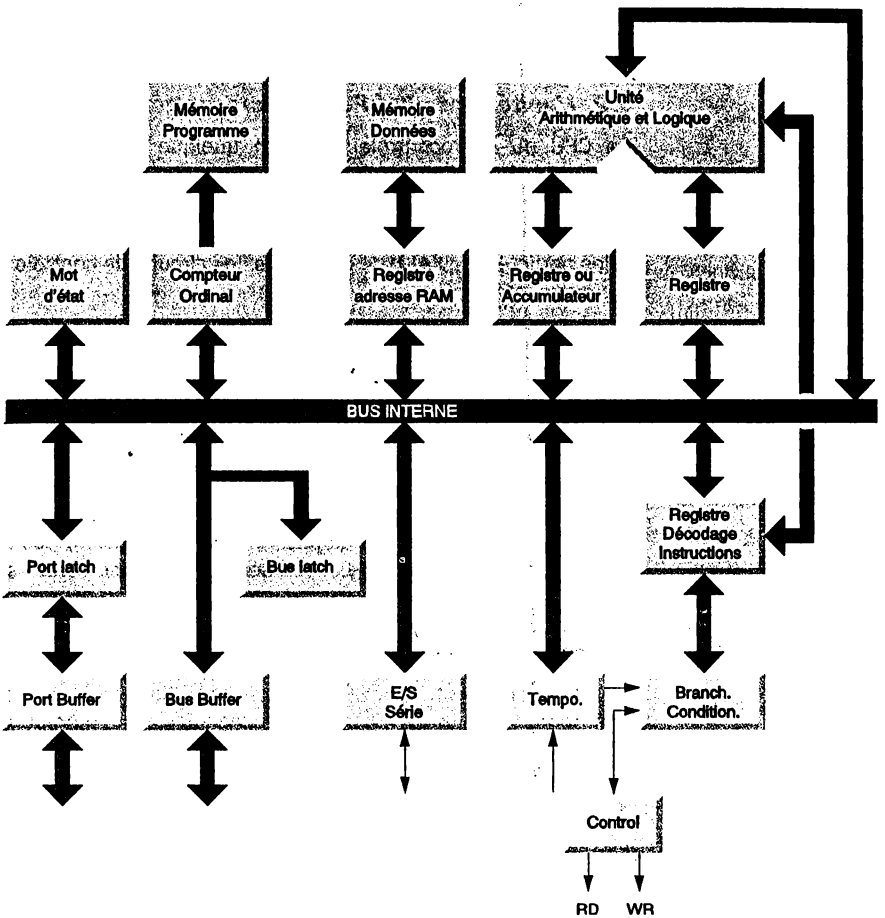



Fig. 4.2
Microcontrôleur de base.

### 1.1 Ports d'entrée/sortie

Les ports d'entrée/sortie contrôlent les événements en provenance du système dans lequel est intégré le microcontrôleur :
- interrupteurs,
- capteurs,
- clavier, etc.

Les ports de sortie transmettent les résultats aux organes du système :
- afficheurs à diodes électroluminescentes,
- relais,
- triacs,
- moteurs pas à pas.

Dans certains microcontrôleurs, les ports d'entrée/sorties sont lus et écrits de la même façon que la mémoire de données.

### 1.2 L'unité centrale CPU du microcontrôleur de base

La CPU du microcontrôleur est en quelque sorte son centre nerveux puisqu'elle contrôle et coordonne les activités des différentes unités du microcontrôleur.

La CPU comporte quatre sous-ensembles spécifiques ayant des rôles particuliers : l'**ALU**, les registres, les circuits de contrôle et la logique d'interruption.

### 1.3 L'unité logique et arithmétique : ALU

L'ALU a la tâche d'effectuer toutes les opérations logiques et arithmétiques. En général, une ALU est capable d'additionner, soustraire, incrémenter et décrémenter toute quantité exprimée sur 8 bits, ainsi que toute opération logique sur 8 bits.

Dans les microcontrôleurs 8 bits les plus simples, l'ALU ne peut réaliser directement une multiplication ou une division. Ces fonctions sont alors réalisées par des séquences d'additions - soustractions et décalages.

Les microcontrôleurs plus performants, comme le 8051, permettent ces fonctions de multiplication - division.

L'ALU du microcontrôleur 8051 est capable d'effectuer toutes les opérations booléennes ainsi que des décalages et la manipulation de bits.

### 1.4 Les registres

Les registres sont des cellules de stockage temporaire 8 bits. Certains de ces registres ont des utilisations spécifiques : ACCU, registres de travail, compteur ordinal PC, registre d'état et registre d'instruction.

*a) L'accumulateur ACCU*

C'est un registre qui reçoit le résultat issu de l 'ALU. Dans de nombreux cas, il contient souvent un opérande qui sera remplacé par un résultat.

*b) Les registres de travail*

Accessibles aisément et rapidement, ils mémorisent les résultats intermédiaires. Ils sont utilisés pour des accès fréquents.

*c) Le compteur ordinal PC*

Il contient l'adresse mémoire de l'instruction à exécuter. Il est mis à jour par la CPU à la fin de chaque instruction. Son contenu sera alors l'adresse de la prochaine instruction à exécuter.

*d) Le registre d'état PSW*

Le registre d'état PSW - Program Status Word - contient un mot qui définit l'état du microcontrôleur. En général, il contient les informations suivantes :
- banque de registres de travail,
- retenue,
- retenue auxiliaire,
- dépassement,
- bits de travail, flags, définis par l'utilisateur.

Un flag est simplement un indicateur quelconque (ce terme n'est, en général, pas traduit).

Il est utile de l'observer en cas de problème puisqu'il permet de savoir exactement ce qui s'est passé et son analyse facilite un diagnostic.

*e) Le registre d'instruction*

Ce registre contient le code opération de l'instruction en cours. Ce code est décodé et interprété par le circuit de contrôle.

Le code opération est lu dans la mémoire programme adressée par le compteur ordinal PC.

## 2 Les mémoires du microcontrôleur

### *2.1 Mémoire programme*

La mémoire programme est une mémoire à lecture seule et peut être de l'un des quatre types suivants :
- ROM : Read Only Memory (code gravé par le fabricant de semi-conducteurs) ;
- PROM : Programmable Read Only Memory (code gravé par l'utilisateur) ;
- EPROM : Erasable Programmable Read Only Memory (code gravé par l'utilisateur et effaçable par ultraviolets) ;
- EEPROM : Electrically Erasable Read Only Memory (code gravé par l'utilisateur et effaçable électriquement).

Les mémoires du type ROM et PROM, qui ne sont pas effaçables, sont utilisées lorsque la phase de développement est terminée et qu'aucune modification n'est à envisager.

Les mémoires EPROM et EEPROM, effaçables, sont utilisées pendant les phases de mise au point du programme.

### *2.2 Mémoire de données*

Les mémoires de données sont des mémoires à écriture/lecture et peuvent être de l'un des deux types suivants :
- RAM (Random Access Memory),
- EEPROM.

Dans certains microcontrôleurs, les ports d'entrée/sorties sont lus et écrits de la même façon que la mémoire de données.

### 1.2 L'unité centrale CPU du microcontrôleur de base

La CPU du microcontrôleur est en quelque sorte son centre nerveux puisqu'elle contrôle et coordonne les activités des différentes unités du microcontrôleur.

La CPU comporte quatre sous-ensembles spécifiques ayant des rôles particuliers : l'**ALU**, les registres, les circuits de contrôle et la logique d'interruption.

### 1.3 L'unité logique et arithmétique : ALU

L'ALU a la tâche d'effectuer toutes les opérations logiques et arithmétiques. En général, une ALU est capable d'additionner, soustraire, incrémenter et décrémenter toute quantité exprimée sur 8 bits, ainsi que toute opération logique sur 8 bits.

Dans les microcontrôleurs 8 bits les plus simples, l'ALU ne peut réaliser directement une multiplication ou une division. Ces fonctions sont alors réalisées par des séquences d'additions - soustractions et décalages.

Les microcontrôleurs plus performants, comme le 8051, permettent ces fonctions de multiplication - division.

L'ALU du microcontrôleur 8051 est capable d'effectuer toutes les opérations booléennes ainsi que des décalages et la manipulation de bits.

### 1.4 Les registres

Les registres sont des cellules de stockage temporaire 8 bits. Certains de ces registres ont des utilisations spécifiques : ACCU, registres de travail, compteur ordinal PC, registre d'état et registre d'instruction.

*a) L'accumulateur ACCU*

C'est un registre qui reçoit le résultat issu de l 'ALU. Dans de nombreux cas, il contient souvent un opérande qui sera remplacé par un résultat.

*b) Les registres de travail*

Accessibles aisément et rapidement, ils mémorisent les résultats intermédiaires. Ils sont utilisés pour des accès fréquents.

*c) Le compteur ordinal PC*

Il contient l'adresse mémoire de l'instruction à exécuter. Il est mis à jour par la CPU à la fin de chaque instruction. Son contenu sera alors l'adresse de la prochaine instruction à exécuter.

*d) Le registre d'état PSW*

Le registre d'état PSW - Program Status Word - contient un mot qui définit l'état du microcontrôleur. En général, il contient les informations suivantes :
- banque de registres de travail,
- retenue,
- retenue auxiliaire,
- dépassement,
- bits de travail, flags, définis par l'utilisateur.

Un flag est simplement un indicateur quelconque (ce terme n'est, en général, pas traduit).

Il est utile de l'observer en cas de problème puisqu'il permet de savoir exactement ce qui s'est passé et son analyse facilite un diagnostic.

*e) Le registre d'instruction*

Ce registre contient le code opération de l'instruction en cours. Ce code est décodé et interprété par le circuit de contrôle.

Le code opération est lu dans la mémoire programme adressée par le compteur ordinal PC.

## 2 Les mémoires du microcontrôleur

### *2.1 Mémoire programme*

La mémoire programme est une mémoire à lecture seule et peut être de l'un des quatre types suivants :
- ROM : Read Only Memory (code gravé par le fabricant de semi-conducteurs) ;
- PROM : Programmable Read Only Memory (code gravé par l'utilisateur) ;
- EPROM : Erasable Programmable Read Only Memory (code gravé par l'utilisateur et effaçable par ultraviolets) ;
- EEPROM : Electrically Erasable Read Only Memory (code gravé par l'utilisateur et effaçable électriquement).

Les mémoires du type ROM et PROM, qui ne sont pas effaçables, sont utilisées lorsque la phase de développement est terminée et qu'aucune modification n'est à envisager.

Les mémoires EPROM et EEPROM, effaçables, sont utilisées pendant les phases de mise au point du programme.

### *2.2 Mémoire de données*

Les mémoires de données sont des mémoires à écriture/lecture et peuvent être de l'un des deux types suivants :
- RAM (Random Access Memory),
- EEPROM.

## 3 Caractéristiques du microcontrôleur 8 bits 8051 (figure 4.3)

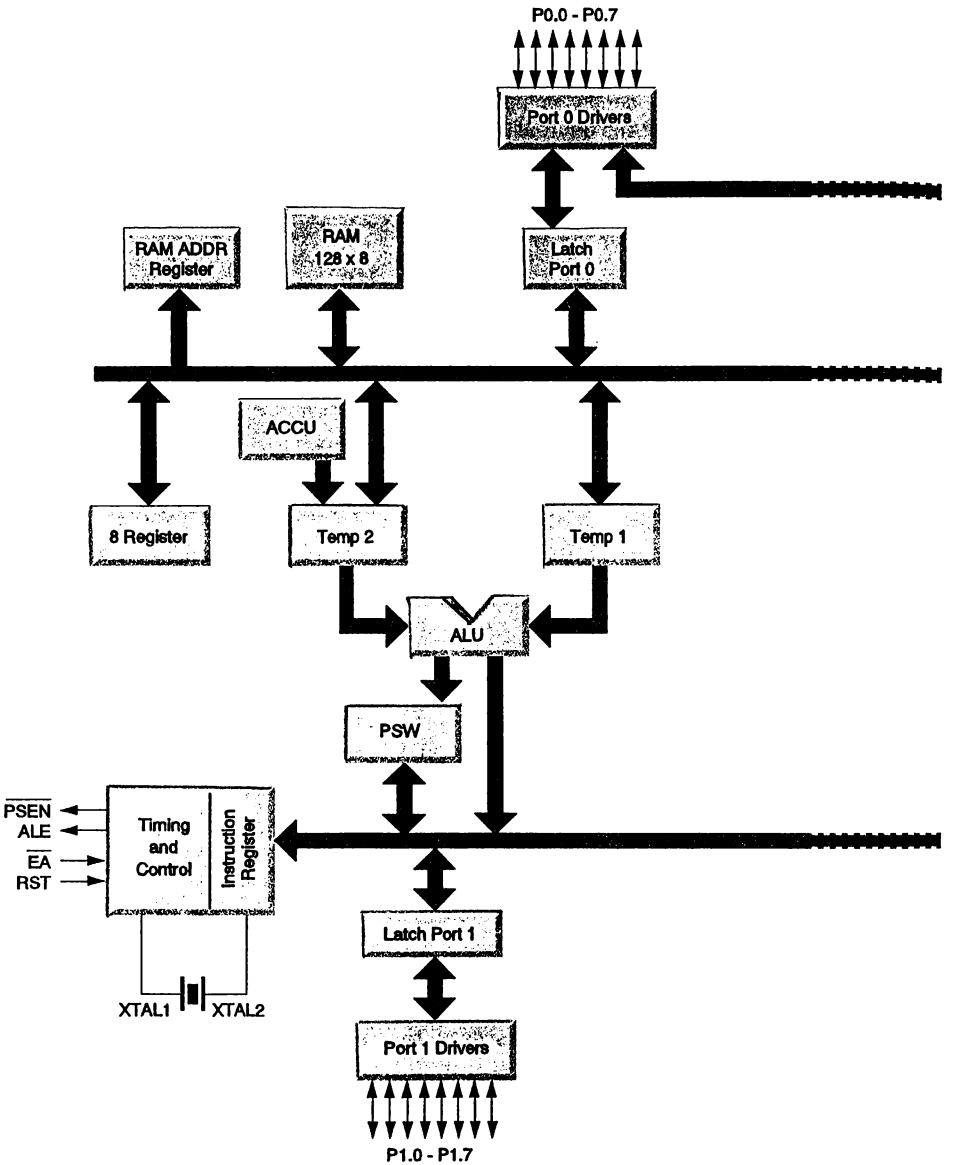



Fig. 4.3
Architecture interne du microcontrôleur 8051.

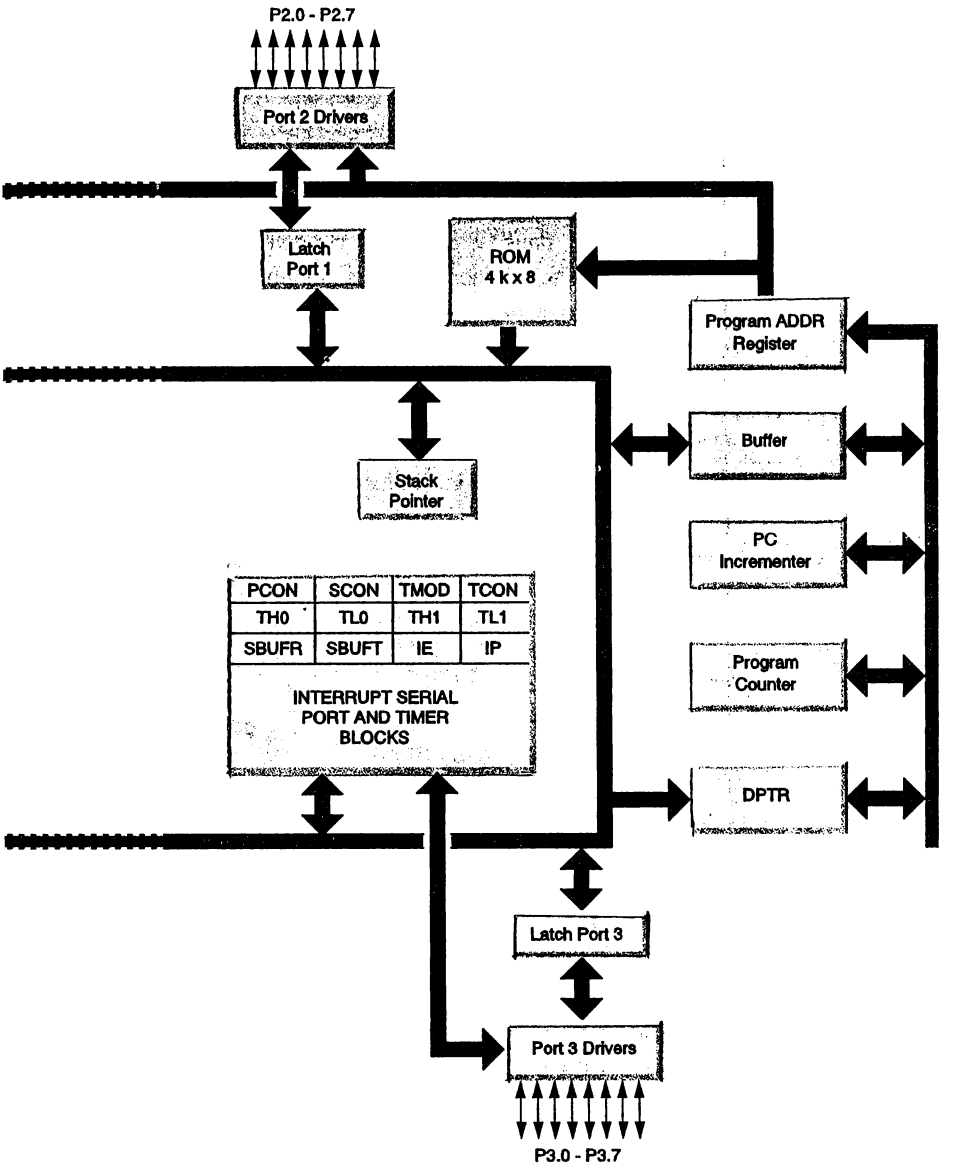

P2.0 - P2.7
Port 2 Drivers
Latch
Port 1
ROM
4 k x 8
Program ADDR
Register
Buffer
Stack
Pointer
PC
Incrementer
PCON
SCON
TMOD
TCON
TH0
TL0
TH1
TL1
SBUFR
SBUFT
IE
IP
INTERRUPT SERIAL
PORT AND TIMER
BLOCKS
Program
Counter
DPTR
Latch Port 3
Port 3 Drivers
P3.0 - P3.7

Fig. 4.3 (suite)

Les caractéristiques principales de ce microcontrôleur sont :
- CPU 8 bits,
- 4 kilo-octets de mémoire ROM,
- 128 octets de mémoire RAM,
- 21 registres spéciaux SFR,
- 32 lignes d'entrée/sorties,
- 64 kilo-octets d'espace d'adressage ROM,
- 64 kilo-octets d'espace d'adressage RAM,
- 2 compteurs temporisateurs 16 bits,
- 5 sources d'interruption,
- une entrée/sortie série full duplex.

Dans une liaison full duplex, les données peuvent circuler simultanément, dans les deux sens, sur deux médias différents.

Tous ces sous-ensembles sont contenus dans un circuit intégré dont le brochage est représenté à la figure 4.4.

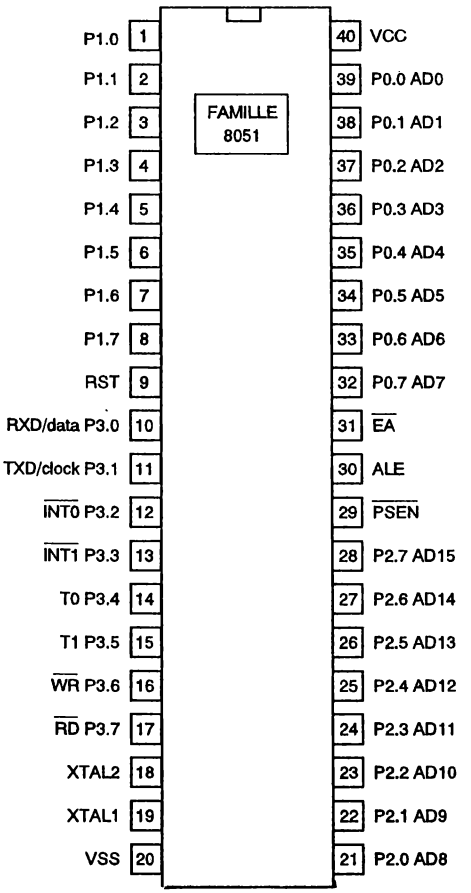



Fig. 4.4
Circuit intégré 8051.

## 4 Configuration hardware du microcontrôleur

Il existe quatre configurations standards du microcontrôleur 8051 avec ou sans RAM ou ROM externe. Ces configurations seront données dans l'ordre de complexité croissante.

### *4.1 Microcontrôleur 8051 ou 8751*

Cette configuration est représentée à la figure 4.5. Il s'agit de la configuration minimum.

La broche EA/VP est utilisée pour indiquer à la CPU la situation de la mémoire programme. Si EA/VP est à l'état haut, le programme est dans la ROM interne. Si EA/VP est à l'état bas, la mémoire programme est externe.

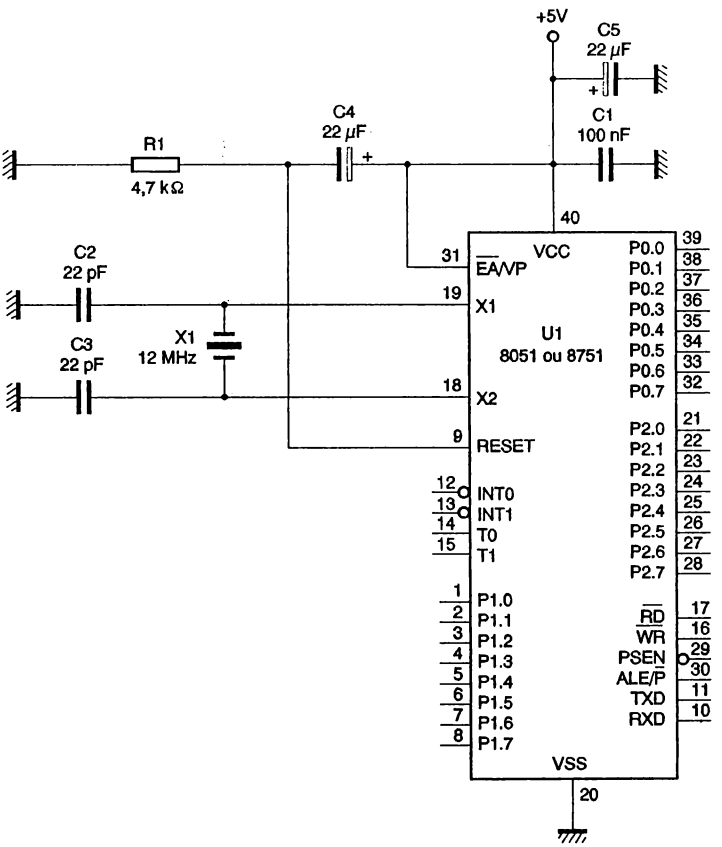



Fig. 4.5
Configuration minimale 8051.

Dans le cas de la figure 4.5, EA/VP est à l'état haut et le programme utilise la mémoire interne. Le programme ne peut dépasser la capacité interne 4 kilo-octets ni utiliser plus de 128 octets de mémoire vive.

Des versions plus élaborées de ce microcontrôleur ont des capacités ROM de 16 kilo-octets et RAM de 256 octets.

Pendant la phase de mise au point ou de test du programme, le microcontrôleur est un 8751 effaçable par ultraviolet. Dès que cette phase est achevée, le microcontrôleur effaçable - en général coûteux - est remplacé, soit par un 8751 OTP (One Time Programming) pour les petites et moyennes séries, soit par un 8051 pour lequel le code est gravé définitivement par le fabricant dans la ROM interne.

Cette configuration présente de nombreux avantages, dont l'utilisation d'un minimum de composants, donc un moindre coût.

Toutes les ressources du microcontrôleur sont préservées. Il est alors possible d'utiliser les quatre ports 8 bits du µC. Ceci n'est pas possible dans les cas suivants.

Il faut finalement préciser que le code gravé dans le microcontrôleur peut être rendu inviolable.

Une sécurité interdit toute lecture, donc recopie du programme interne.

*Définition des broches du microcontrôleur famille 8051*

$V_{ss}$ 0 Volt

$V_{cc}$ 5 Volts

PORT 0 : Le port 0 est un port 8 bits quasi-bidirectionnel (collecteur ouvert).
La structure du PORT 0 est très particulière : on peut par exemple écrire directement une valeur en sortie ou mémoriser cette valeur dans une bascule.
Cette caractéristique vaut au PORT 0 le nom de **quasi-bidirectionnel**.
Il est multiplexé avec les poids faibles d'adresse lors d'un accès à une mémoire externe. Le schéma, donnant les connexions avec une mémoire externe, montre clairement ce multiplexage.

PORT 1 : Le port 1 est un port 8 bits quasi-bidirectionnel (collecteur ouvert).

PORT 2 : Le port 2 est un port 8 bits quasi-bidirectionnel (collecteur ouvert).
Il est multiplexé avec les poids forts d'adresse lors d'un accès à une mémoire externe.

PORT 3 : Le port 3 est un port 8 bits quasi-bidirectionnel (collecteur ouvert).

Il peut être utilisé à d'autres fonctions :
- P3.0 RXD (entrée série),
- P3.1 TXD (sortie série),
- P3.2 $\overline{INT0}$ ( interruption externe),
- P3.3 $\overline{INT1}$ (interruption externe),
- P3.4 T0 (entrée du compteur/temporisateur0),
- P3.5 T1 (entrée du compteur/temporisateur 1),
- P3.6 $\overline{WR}$ (signal d'écriture de la mémoire RAM externe),
- P3.7 $\overline{RD}$ (signal de lecture de la mémoire RAM externe).

RST : Signal de remise à zéro du microcontrôleur.

ALE : Signal de démultiplexage du port 0 (Address Latch Enable).

PSEN : Signal de lecture de la mémoire de programme externe (Program Store Enable).

XTAL1
XTAL2 : Entrées quartz ou horloge.

$\overline{EA}$ : Quand EA est actif, le CPU exécute seulement le programme contenu dans la ROM externe.

### *4.2 Microcontrôleur avec RAM externe*

Dans le cas de la figure 4.6, une RAM externe de 8 K. 8 est connectée au microcontrôleur.

Cette configuration est adoptée si la capacité de la RAM interne est insuffisante. Ceci ne représente pas la capacité RAM maximale 64 K. 8.

Les accès à une mémoire de données externe utilisent les signaux $\overline{RD}$ et $\overline{WD}$ respectivement pour la lecture et l'écriture.

Dans ce cas, les ports d'entrée/sortie P0 et P2 sont utilisés pour véhiculer les adresses et les données transitant entre la mémoire et le microcontrôleur.

Les adresses basses A0-A7 et les données D0-D7 sont multiplexées sur le port P0.

Le démultiplexage adresses/données est effectué grâce au signal ALE.

La fonction d'ALE est donc de mémoriser les poids faibles d'adresse quand ils sont présents sur P0. ALE est actif deux fois à chaque cycle machine, sauf pendant un accès à une mémoire de données externe.

Si un système ne comporte pas de mémoire de données externe, ALE peut être utilisé comme base de temps ayant une fréquence de 1/6 de la fréquence de l'oscillateur.

L'adjonction d'une RAM externe consomme une partie des ressources du microcontrôleur. Les ports P0, P2 et une partie du port P3 – $\overline{RD}$ et $\overline{WR}$ – ne peuvent plus être utilisés comme entrées/sorties banalisées.

Le programme est toujours interne et les mêmes considérations que dans la structure précédente peuvent s'appliquer.

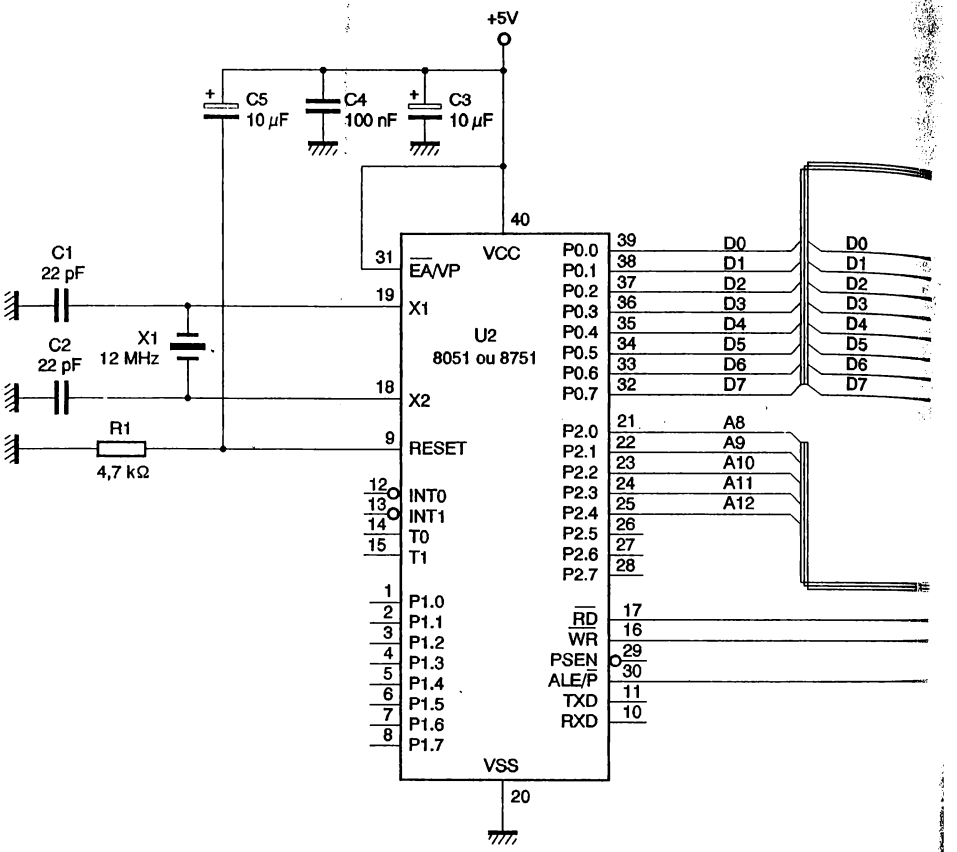



Fig. 4.6
Microcontrôleur avec RAM externe.

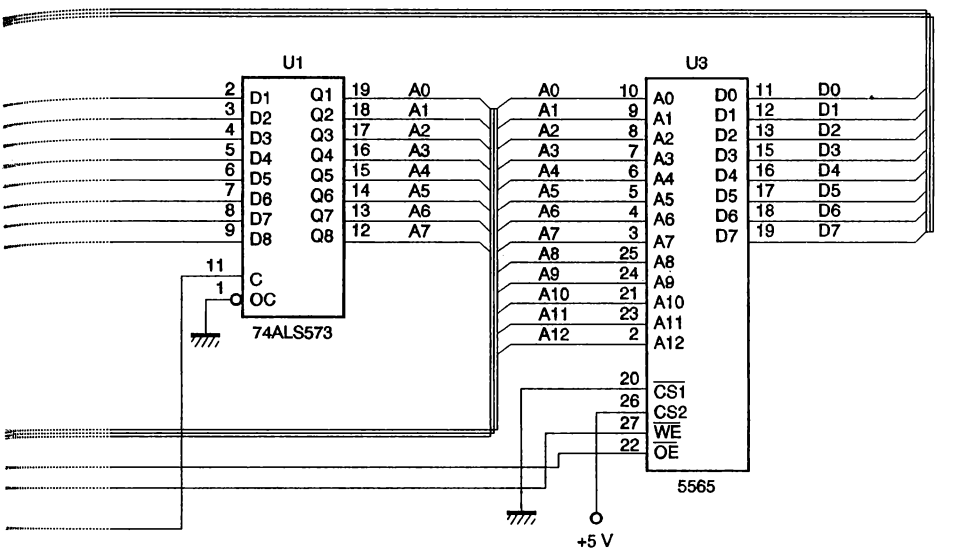



Fig. 4.6 (suite)

### *4.3 Microcontrôleur avec EPROM externe*

Dans la famille 8051, la taille maximale du programme pouvant loger dans la ROM interne est de 16 kilo-octets. Si le programme a une taille supérieure, il doit être logé dans une mémoire externe. Le schéma de la figure 4.7 rend compte de la configuration à adopter.

Dans ce cas, une mémoire de 32 kilo-octets, qui ne représente pas la taille maximale, est connectée au microcontrôleur de la même manière que la RAM dans l'exemple précédent.

La broche EA/VP est à l'état bas, indiquant au processeur que le programme se situe en mémoire externe.

Les accès à une mémoire programme externe utilisent le signal $\overline{PSEN}$ comme signal de lecture de la mémoire. Les accès à la mémoire de programme ne se font qu'en lecture.

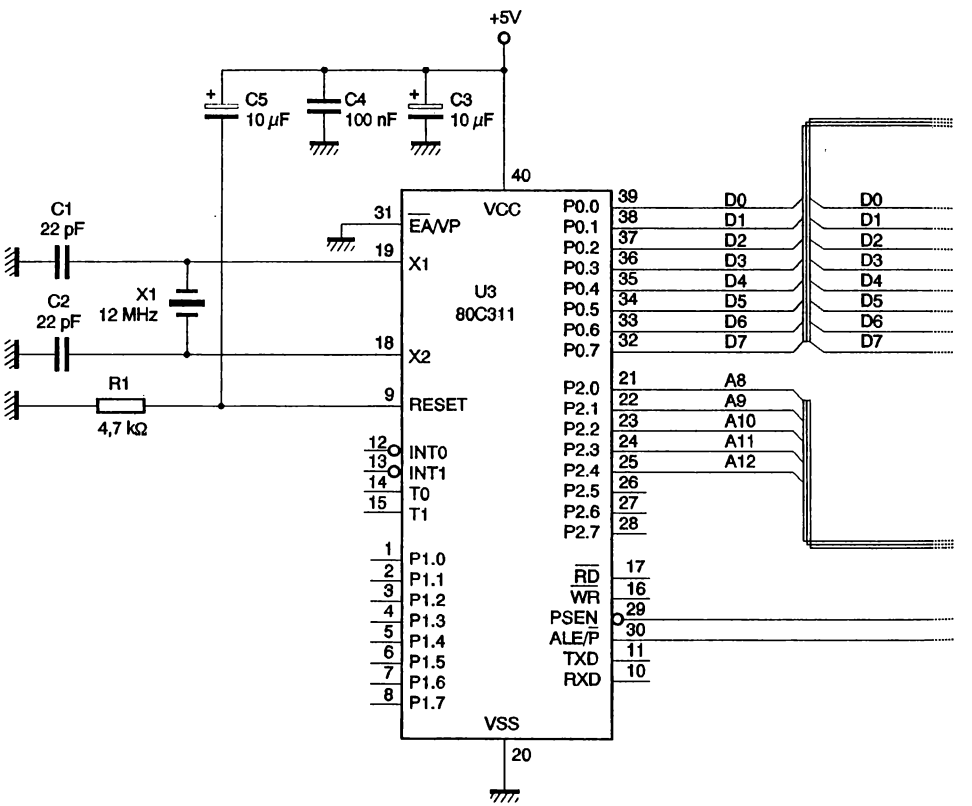



Fig. 4.7
Microcontrôleur avec mémoire programme externe.

Le microcontrôleur ne comporte plus de mémoire programme interne, il est alors baptisé 8031. Sa mémoire RAM interne reste inchangée.

Contrairement aux deux exemples précédents, le programme est externe.

Il est donc impossible de se protéger d'une éventuelle copie.

Comme précédemment, une partie des ressources du microcontrôleur est affectée à la gestion de la mémoire.

En contrepartie, cette configuration est la plus économique (moins de 30 Francs) pour l'ensemble du schéma de la figure 4.7.

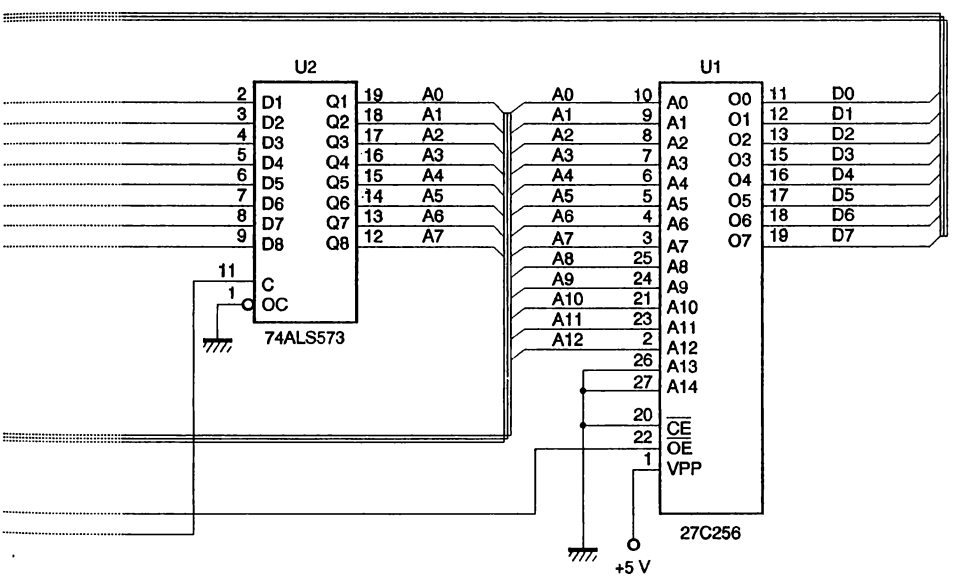



Fig. 4.7 (suite)

### *4.4 Microcontrôleur avec RAM et ROM externes*

La configuration de la figure 4.8 est la configuration hardware la plus complexe. Elle regroupe en fait les schémas des figures 4.6 et 4.7.

La broche EA/VP est au niveau bas, indiquant que le programme est en mémoire externe.

La mémoire RAM est gérée par les signaux $\overline{RD}$ et $\overline{WR}$. La mémoire EPROM est gérée par les signaux $\overline{RD}$ et $\overline{PSEN}$.

Cette configuration matérielle est la plus coûteuse aussi bien en terme de composants qu'en circuit imprimé, câblage et maintenance. Elle est naturellement réservée aux programmes importants devant manipuler un grand nombre de données en RAM.

On retiendra que la mémoire programme peut être interne, externe ou interne et externe comme le montre le synoptique de la figure 4.9.

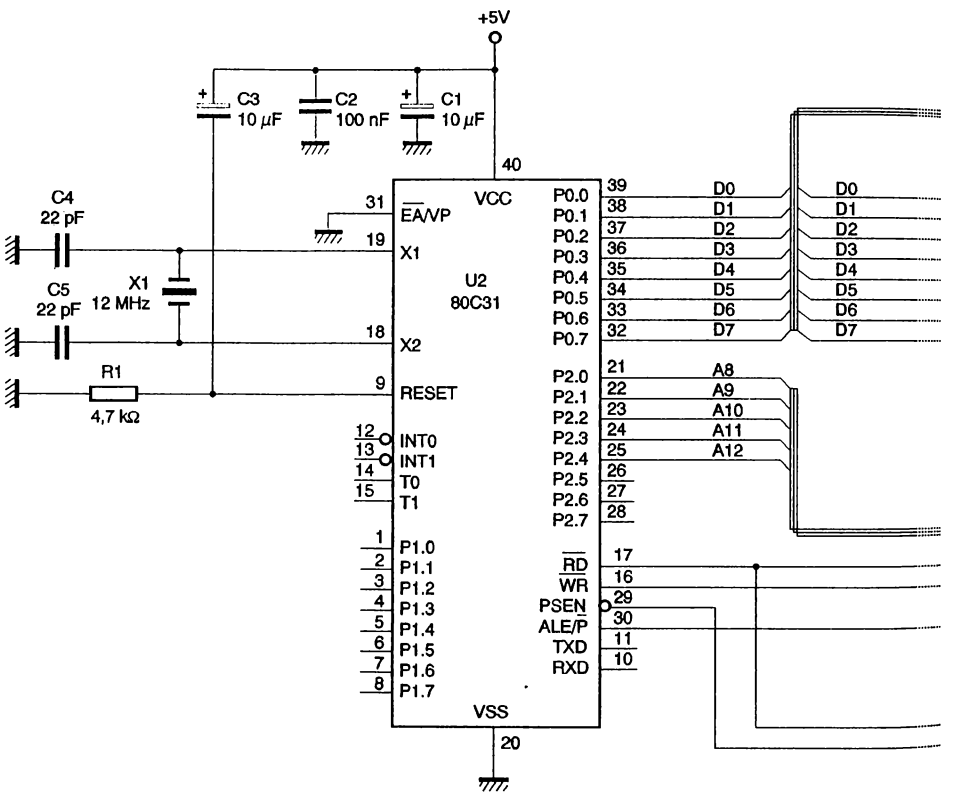



Fig. 4.8
Microcontrôleur avec RAM et ROM externes.

Dans le cas de la mémoire programme mixte, les quatre premiers kilo-octets sont internes au microcontrôleur, les suivants sont en mémoire externe.

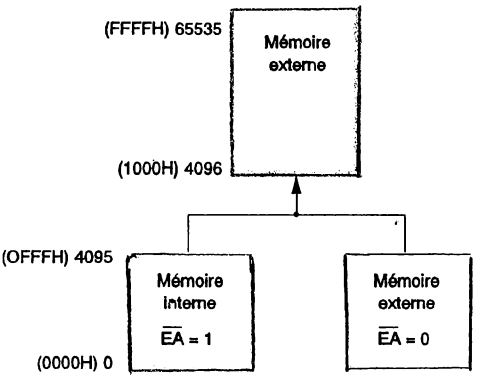



Fig. 4.9
Configuration de la mémoire programme.

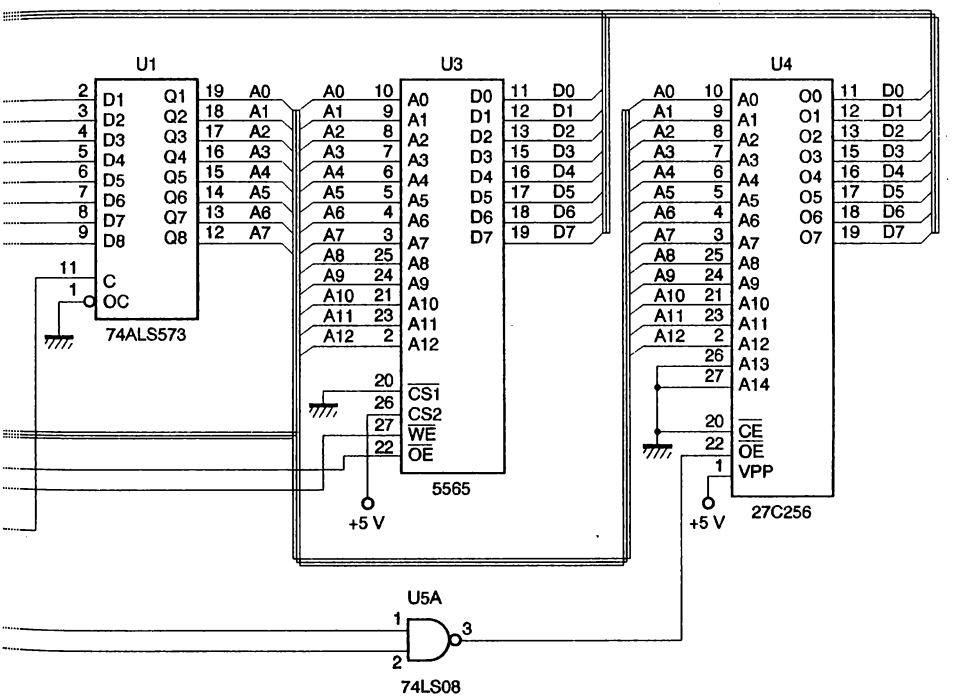



Fig. 4.8 (suite)

## 5 La mémoire RAM en détail

La mémoire de données comprend :
- 128 octets de RAM interne,
- 21 registres spéciaux SFR,
- 64 kilo-octets de RAM externe.

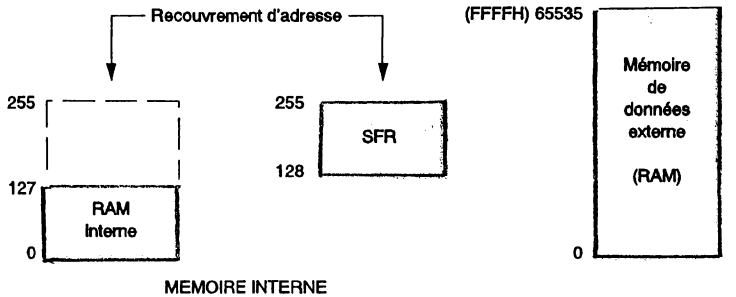



Fig. 4.10
Organisation de la mémoire RAM.

L'organisation et l'adressage de la mémoire sont représentés à la figure 4.10.

La RAM interne utilise 8 bits d'adresse – 256 octets max –, la RAM externe peut utiliser 8 ou 16 bits d'adresse.

Les 32 premiers octets de la RAM interne – de l'adresse 00H à 1FH – sont divisés en quatre banques de registres. Chaque banque est constituée de 8 octets.
Les bits des 16 octets suivants – de l'adresse 20H à 2FH – sont individuellement adressables ainsi que les bits de 11 registres spéciaux SFR.

## 6 Les registres spéciaux SFR

La table des registres spéciaux est donnée à la figure 4.11 :
- ACC (accumulateur) ;
- B : le registre B est utilisé par les instructions de multiplication et de division ;
- SP (pointeur de pile ou Stack Pointer). SP est un registre de 8 bits. La pile peut être à n'importe quelle adresse de la RAM interne.

Lors de l'exécution d'un PUSH ou d'un CALL, SP est incrémenté avant que la donnée soit stockée dans la pile. À l'initialisation du microcontrôleur, SP prend la valeur 07H ;
- DPTR (pointeur de données ou Data Pointer). DPTR est un registre de 16 bits formé de 2 octets, DPH pour les poids forts et DPL pour les poids faibles. Il peut pointer sur la mémoire programme et sur la RAM externe ;
- P0. P1, P2 et P3.

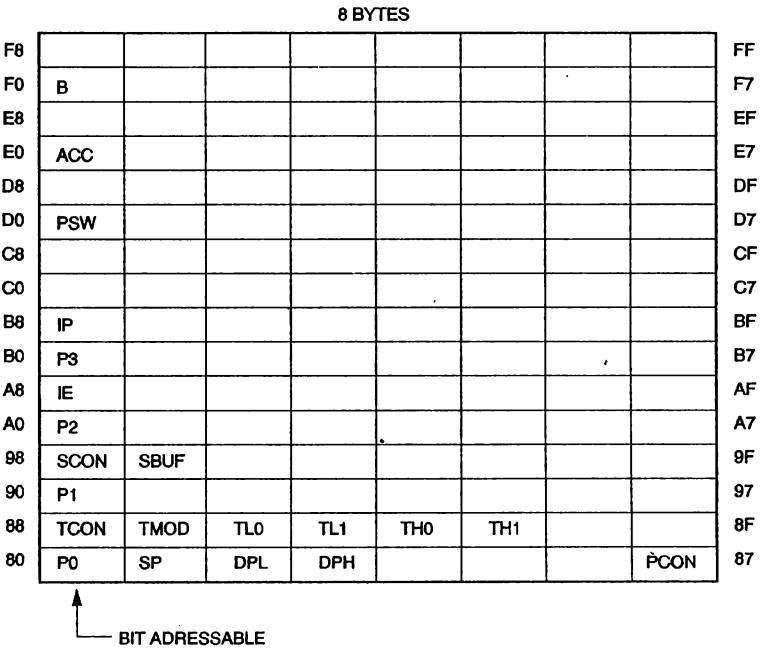



Fig. 4.11
Table des registres spéciaux.

Ces quatre registres fournissent les 32 lignes d'entrées/sorties. P0 et P2 sont utilisés pour accéder aux mémoires externes. P2 fournit les adresses de poids forts, P0 les adresses de poids faibles. P1 est aussi multiplexé dans le temps avec l'octet à lire ou à écrire. P3 peut avoir d'autres fonctions :
P3.0 RXD (entrée série),
P3.1 TXD (sortie série ),
P3.2 $\overline{INT0}$ (interruption externe),
P3.3 $\overline{INT1}$ (interruption externe),
P3.4 T0 (entrée du compteur /temporisateur 0),
P3.5 T1 (entrée du compteur/temporisateur 1),
P3.6 $\overline{WR}$ (signal d'écriture de la mémoire RAM externe),
P3.7 $\overline{RD}$ (signal de lecture de la mémoire RAM externe) ;

– SBUF (tampon de données série ou Serial Data Buffer). SBUF est composé de 2 registres. Les données à transmettre via le port série sont écrites dans SBUF. Les données reçues du port série sont lues dans SBUF ;

– IP, IE,TMOD, SCON, PCON.
IP niveau de priorité des interruptions (Interrupt Priority),
IE validation des interruptions (Interrupt Enable),

TMODE mode de fonctionnement des compteurs/temporisateurs (Timer/counter Mode),
TCON contrôle des compteurs /temporisateurs (Timer/counter Mode),
SCON contrôle du port série (Serial Control),
PCON contrôle de l'alimentation (Power Control) ;

– autres SFR :
PSW Program Status Word (mot d'état),
TH0 timer/counter 0 (poids forts),
TL0 timer/counter (poids faibles),
TH1 timer/counter 1 (poids forts),
TL1 timer/counter (poids faibles).

## 7 Circuit d'horloge et oscillateur

L'oscillateur interne peut être utilisé en connectant directement un quartz sur les broches XTAL1 et XTAL2.

Une horloge extérieure peut être connectée à la broche XTAL2, la broche XTAL1 étant reliée à la masse.

La fréquence de l'horloge est divisée par 2, ainsi le rapport cyclique 1/2 n'est pas nécessaire.

Le diviseur par 2 génère deux phases d'horloge.

La phase 1 est active durant la première moitié de chaque période d'horloge, la phase 2 est active durant la seconde moitié de chaque période d'horloge.

Pour les versions standards des microcontrôleurs 8051, la fréquence maximale de l'oscillateur vaut 12 MHz. Cette fréquence a une incidence directe sur les performances et la rapidité des calculs. Il existe des versions plus performantes où la fréquence horloge peut atteindre 33 MHz.

## 8 Timing du CPU

Un cycle machine est divisé en 6 états ou 12 périodes horloge (voir figure 4.12) ; chaque état est divisé en 2 phases correspondant aux deux phases du signal d'horloge.

Normalement les opérations arithmétiques et logiques sont exécutées pendant la phase 1 et les opérations internes de transfert de registre à registre pendant la phase 2.

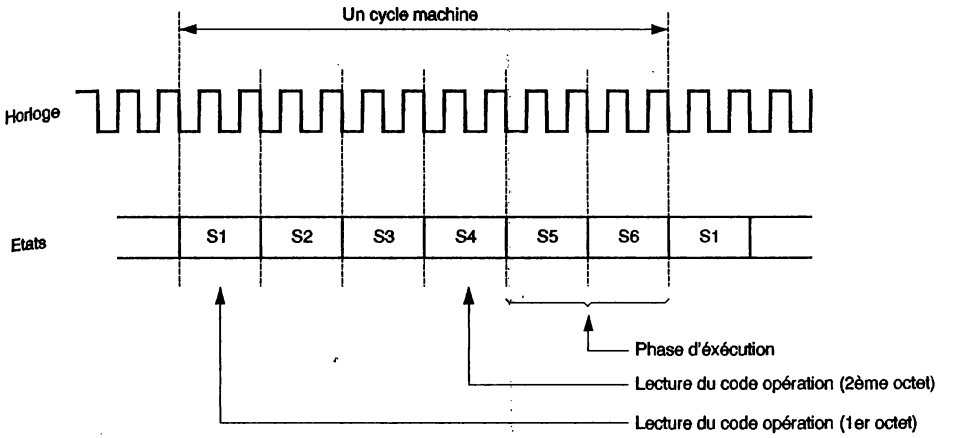



Fig. 4.12
Cycle machine pour un code opération à 2 octets.

## 9 Les opérations d'entrées-sorties

Chaque ligne d'entrée/sortie peut être utilisée indépendamment en entrée ou en sortie. Quand une ligne doit être utilisée en entrée, le latch correspondant doit contenir 1.

Les ports 1,2 et 3 sortent sur un collecteur de transistor avec une résistance de rappel au + 5 V. Quand ce transistor est bloqué, la ligne peut être mise à zéro par une source extérieure (fonction entrée).

Pour le port 0, la résistance de rappel est remplacée par un transistor. Ce transistor est bloqué sauf lors d'accès à une mémoire extérieure. Si les deux transistors sont bloqués, la ligne est flottante.

Les instructions qui lisent le latch ont aussi la possibilité de le modifier. Ces instructions sont appelées des instructions de « read - modify - write » :

| | |
|---|---|
| ANL | : ET logique (exemple : ANL P1, A) |
| ORL | : OU logique (exemple : ORL P2, A) |
| XRL | : OU exclusif (exemple : XRL P3, A) |
| JBC | : Branchement si bit = 1 et RAZ bit (exemple : JBC P1, 1, LABEL) |
| CPL | : Complémentation d'un bit (exemple : CPL P3, 0) |
| INC | : Incrémentation (exemple : INC P2) |
| DEC | : Décrémentation (exemple : DEC P2) |
| DJNZ | : Décrémentation et branchement si différent de zéro (exemple : DJNZ P2, LABEL) |
| MOV PX, Y, C | : Transfert du bit de retenue dans le bit Y du port X |
| CLR PX, Y | : Remise à zéro du bit Y du port X |
| SET PX, Y | : Mise à 1 du bit Y du port X |

Les trois dernières instructions ne sont pas véritablement des instructions de « read - modify - write ».

Toutes ces instructions lisent la valeur du latch et non la valeur de la broche de sortie.

MOV A, P1 ne lit pas la valeur du latch mais la valeur de la broche de sortie.

## 10 Écriture d'un port

Lors de l'exécution d'une instruction de modification d'un latch de port, la nouvelle valeur est chargée dans le latch pendant la phase 2 de l'état (S6P2) du dernier cycle de l'instruction.

Si la valeur d'un port de sortie doit passer de l'état 0 à l'état 1, une résistance additionnelle est mise en parallèle sur la résistance de sortie pendant les temps S1P1 et S1P2.

## 11 Compteur/temporisateur (Timer/event Counter)

Le 8051 possède deux registres de 16 bits, Timer 0 et Timer 1, qui peuvent être utilisés en temporisateur ou en compteur d'événements (timer or event counter). Leurs modes de fonctionnement et leurs configurations s'effectuent à partir du SFR TMOD.

En fonction Timer, le registre est incrémenté tous les cycles machine. Sa fréquence de comptage est donc de 1/12 de la fréquence de l'oscillateur.

En fonction compteur, le registre est incrémenté à chaque transition 1-0 sur son entrée respective T0 ou T1. Dans cette fonction, les entrées T0 ou T1 sont échantillonnées à chaque cycle machine au temps S5P2. Il faudra donc un maximum de 24 périodes de l'oscillateur pour être reconnu. Chaque niveau 1 ou 0 devra être maintenu sur T0 ou T1 au moins pendant un cycle machine complet.

En plus de la fonction Timer ou Counter, chaque registre possède quatre modes opératoires. Les modes 0,1 et 2 sont les mêmes pour les deux registres ; le fonctionnement en mode 3 est différent.

### *11.1 Modes 0 et 1*

Dans le mode 0, le registre Timer est configuré en 13 bits.

Quand tous les bits passent de 1 à 0, il positionne TF1 à 1. L'entrée compteur est validée quand TR1 = 1 et GATE = 0 ou $\overline{INT1}$ = 1.

TR1 est un bit de contrôle du SFR TCON, GATE est un bit de contrôle du SFR TMOD.

Les trois bits de poids forts de TL1 sont ignorés.

Le mode 0 est identique pour les deux timers. Seuls les bits de contrôle sont différents.

Le mode 1 a le même fonctionnement que le mode 0 avec l'exception suivante : le comptage s'effectue sur 16 bits au lieu de 13 bits.

### 11.2 Mode 2

Le mode 2 configure le registre Timer en 8 bits avec rechargement automatique quand il déborde.

Quand il y a débordement, TF1 est mis à 1 et TL1 est rechargé avec le contenu de TH1, qui reste inchangé.

Le mode 2 est le même pour les deux timers.

### 11.3 Mode 3

Si le timer est mis en mode 3, TL0 et TH0 deviennent deux compteurs séparés. TL0 utilise tous les bits de contrôle du timer 0. TH0 n'a que la fonction timer (comptage de cycles machine) et utilise TR1 et TF1 du timer 1.

Normalement on ne configure pas le timer en mode 3 quand le timer 1 est utilisé (sauf pour des applications spécifiques).

### 11.4 Contrôle et mot d'état des « timer/event counters »

Deux SFR, TMOD et TCON, sont utilisés pour définir les modes opératoires et contrôler les fonctions des « timer/event counters ».

Quand une instruction modifie le contenu de TMOD ou TCON, le changement intervient au temps S1P1 du premier cycle de l'instruction suivante.

| | Timer 1 | | | | Timer 0 | | | |
|---|---|---|---|---|---|---|---|---|
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | GATE | C / $\overline{T}$ | M1 | M0 | GATE | C / $\overline{T}$ | M1 | M0 |

– C/T : choix de la fonction Timer ou compteur (en compteur, le compteur compte les transitions négatives sur T0 ou T1).

– GATE : quand GATE = 1, le timer est validé seulement quand $\overline{INTx} = 1$ et TRx = 1. Si GATE =0, le timer est validé par TRx = 1.

| M1 | M0 | Mode | Description |
|---|---|---|---|
| 0 | 0 | 0 | Compteur 13 bits |
| 0 | 1 | 1 | Compteur 16 bits |
| 1 | 0 | 2 | Compteur 8 bits avec rechargement |
| 1 | 1 | 3 | Timer 0 divisé en 2 compteurs 8 bits. |

Tous les bits de TMOD sont remis à zéro à l'initialisation (Reset).

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TF1 | TR1 | TF0 | TR0 | IE1 | IT1 | IE0 | IT0 |

– TF1 : positionné à 1 par hardware quand il y a débordement du Timer et remis à zéro par hardware quand le processeur part en routine d'interruption.

– TR1 : bit de validation de comptage du timer 1. Positionné à 1 ou 0 par programme.

– TF0 : positionné à 1 par hardware quand il y a débordement du timer 0 et remis à zéro par hardware quand le processeur part en routine d'interruption.

– TR1 : bit de validation de comptage du timer 0. Positionné à 1 ou par programme.

– IE1 : si IT1 = 1, ce bit passe à 1 quand une transition 1-0 apparaît sur $\overline{INT1}$.

– IT1 : ce bit détermine si l'interruption externe 1 est active sur une transition ou un niveau. Si IT1 = 1, $\overline{INT1}$ est active sur une transition ; si IT1 = 0, $\overline{INT1}$ est active sur un niveau.

– IE0 : si IT0 = 1, ce bit passe à 1 quand une transition 1-0 apparaît sur $\overline{INT0}$.

– IT0 : ce bit détermine si l'interruption externe 1 est active sur une transition ou un niveau. Si IT0 = 1, $\overline{INT0}$ est active sur une transition ; si IT0 = 0, $\overline{INT0}$ est active sur un niveau.

Tous les bits de TCON sont remis à zéro à l'initialisation (Reset).

## Interface série

C'est une **interface série** « full duplex », ce qui signifie qu'elle peut émettre ou recevoir des données série simultanément. Elle peut recevoir une seconde donnée alors que la première n'a pas encore été lue par le CPU.

Il y a deux registres tampon réservés au port série à l'adresse SFR SBUF. Une écriture dans SBUF charge le registre SBUF de transmission, une lecture de SBUF lit le registre de réception (physiquement différent du premier).

Le port série a quatre modes opératoires, détaillés ci-après.

### *12.1 Mode 0. Réception Transmission synchrone*

8 bits de données série sont transmises ou reçues sur la broche RXD et TXD fournit l'horloge de « shift ». Les données sont émises et reçues, les poids faibles en tête. La vitesse de transmission est fixée au 1/12 de la fréquence de l'oscillateur.

### *12.2 Mode 1. Réception Transmission asynchrone*

10 bits sont transmis sur TXD ou reçus sur RXD : un bit de start (0), 8 bits de données (poids faibles en tête), un neuvième bit programmable, et un bit de stop. La vitesse de transmission est de 1/32 ou 1/64 de la fréquence de l'oscillateur.

### *12.3 Modes 2 et 3. Réception Transmission asynchrone*

Le mode 3 est le même que le mode 2, excepté la vitesse de transmission qui est variable.

En modes 2 et 3, il est possible d'établir une communication entre plusieurs microcontrôleurs.

Dans ces modes, le neuvième bit reçu est écrit dans RB8 du SFR SCON. L'interface série peut être programmée de telle sorte qu'une interruption ne soit générée, après la réception du bit de stop, que si RB8 = 1 ; cette possibilité est offerte en positionnant SM2 du SFR SCON à 1.

La communication multiprocesseurs s'établit comme suit.

Quand un processeur maître veut transmettre un bloc de données à un des processeurs esclaves, il envoie d'abord l'adresse du processeur esclave.

L'octet d'adresse est différencié d'un octet de donnée par RB8 = 1 pour une adresse, et RB8 = 0 pour un octet de donnée.

Avec SM2 à 1, aucun esclave ne sera interrompu par une donnée, par contre une adresse interrompra tous les esclaves.

Chaque esclave examinera l'adresse reçue. Si celle-ci correspond à sa propre adresse, il remettra SM2 à 0 et recevra les données qui lui seront alors transmises.

Les esclaves qui ne seront pas reconnus laisseront SM2 à 1 et ignoreront les données transmises.

## 13 Vitesses de transfert ou Baud Rates (figure 4.13)

En mode 0, le « baud rate » est fixé au 1/12 de la fréquence de l'oscillateur.

En mode 2, le « baud rate » est de 1/64 ou 1/32 de la fréquence de l'oscillateur selon la valeur du bit SMOD du SFR PCON.

Si SMOD = 0 (valeur par défaut au reset), le baud rate est de 1/64 de la fréquence de l'oscillateur ; si SMOD = 1, le « baud rate » est de 1/32 de la fréquence de l'oscillateur.

Le « baud rate » en mode 1 et 3 est déterminé par la fréquence de débordement du timer 1 : Baud rate = (Timer 1 overflow rate)/*n*.

La valeur de *n* est déterminée par la valeur du bit SMOD du SFR PCON. Si SMOD = 0 (valeur par défaut au reset), *n* = 32, si SMOD = 1 alors *n* = 16.

Si l'on utilise le timer en Mode 2 (auto reload) : overflow rate = (count rate)/(256-TH1).

| BAUDE RATE | Fosc | SMOD | TIMER 1 | | |
|---|---|---|---|---|---|
| | | | C/T | MODE | RELOAD VALUE |
| MODE 0 max : 1 MHz | 12 MHz | X | X | X | X |
| MODE 2 max : 375 kHz | 12 MHz | 1 | X | X | X |
| MODE 1,3 max : 62,5 kHz | 11,059 MHz | 1 | 0 | 2 | FFH |
| 19,2 kHz | 11,059 MHz | 0 | 0 | 2 | FDH |
| 9,6 kHz | 11,059 MHz | 0 | 0 | 2 | FDH |
| 4,8 kHz | 11,059 MHz | 0 | 0 | 2 | FAH |
| 2,4 kHz | 11,059 MHz | 0 | 0 | 2 | F4H |
| 1,2 kHz | 11,059 MHz | 0 | 0 | 2 | E8H |
| 137,5 Hz | 11,059 MHz | 0 | 0 | 2 | 1DH |
| 110 Hz | 6 MHz | 0 | 0 | 2 | 72H |
| 110 Hz | 12 MHz | 0 | 0 | 1 | FEEBH |

Fig. 4.13
Vitesse de transmission sur le port-série.

### 13.1 SFR PCON (Serial Port Control Register)

SCON est utilisé pour définir le mode opératoire du port série et pour en contrôler certaines fonctions.

Il reçoit aussi le neuvième bit de donnée (RB8) et contient les flags d'interruption en émission et réception (TI RI).

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SM0 | SM1 | SM2 | REN | TB8 | RB8 | TI | RI |

- SM0 et SM1 définissent le mode opératoire du port série :

| SM0 | SM1 | Mode | Description | Baud Rate |
|---|---|---|---|---|
| 0 | 0 | 0 | Synchrone | Fosc/12 |
| 0 | 1 | 1 | 8-bit UART | Variable |
| 1 | 0 | 2 | 9-bit UART | F/64 F/32 |
| 1 | 1 | 3 | 9-bit UART | Variable |

codage SM0 SM1

- SM2 : autorise le mode multiprocesseurs en modes 2 et 3. Dans ces modes, si RM2 = 1, RI sera activé si le neuvième bit reçu (RB8) est à 0. En mode 1, si SM2 = 1, RI ne sera pas activé si un bit de stop n'est pas reçu. En mode 0, SM2 doit être à 0.

– REN : valide la réception. Mis à 1 par software pour autoriser la réception, remis à 0 par software pour invalider la réception.

– TB8 : neuvième bit de donnée à transmettre en modes 2 et 3. En mode 1, si SM2 = 0, RB8 est le bit de stop reçu. En mode 0, RB8 n'est pas utilisé.

– TI : flag d'interruption en transmission. Il est mis à 1 par hardware à la fin du huitième bit transmis en mode 0 ou au début de la réception d'un bit de stop dans les autres modes.

TI doit être remis à zéro par software.

– RI : flag d'interruption en réception. Il est mis à 1 par hardware à la fin du huitième bit reçu en mode 0 ou au milieu de la réception du bit de stop dans les autres modes (sauf si SM2 = 1 et neuvième bit reçu = 1). RI doit être remis à 0 par software.

Tous les bits de SCON sont remis à zéro par Reset.

Quand une instruction change le contenu de SCON, la nouvelle valeur ne sera chargée qu'au temps S1P1 de la prochaine instruction. Si une transmission est en cours, l'ancienne valeur sera maintenue, la nouvelle valeur sera ignorée.

TI et RI génèrent la même interruption. Avant le traitement de cette interruption, il faut analyser TI et RI pour différencier une interruption transmission d'une interruption réception.

### *13.2 Les registres du port série*

Dans tous les modes, une écriture dans SBUF charge le même registre à décalage de 9 bits.

Les 8 bits de données occupent les 8 premiers bits avec le bit de poids faible en tête (le premier à être émis).

Le neuvième bit est chargé à 1 ou par la valeur de RB8 selon les modes.

Le registre de réception est un registre à décalage de 8 bits en mode 0 et de 9 bits dans les autres modes.

La donnée du registre réception est chargée dans SBUF en même temps que RI est activé. Dans les modes UART, le neuvième bit est chargé dans RB8 de SCON.

RB8 et SCON restent inchangés si SM2 est positionné de telle sorte que la réception soit ignorée.

## 14 Les interruptions

Le 8051 a cinq sources d'interruption, chacune ayant deux niveaux de priorité programmable.

Chaque source peut être individuellement validée ou invalidée en positionnant un bit dans le SFR IE. Toutes les interruptions peuvent être globalement validées ou invalidées.

Chaque source peut être programmée avec un niveau de priorité haut ou bas.

Une interruption de niveau bas peut être interrompue par une interruption de niveau de priorité haut.

Une interruption de niveau de priorité haut ne peut pas être interrompue.

| Source | Description |
|---|---|
| $\overline{INT0}$ | Interruption externe (P3.2) échantillonnée à S5P2 de chaque cycle machine. |
| TIMER 0 | Débordement du timer 0. Active le flag d'interruption TF0. |
| $\overline{INT1}$ | Interruption externe (P3.3) échantillonnée à S5P2 de chaque cycle machine. |
| TIMER 1 | Débordement du timer 1. Active le flag d'interruption TF1. |
| PORT SÉRIE | Fin de transmission ou de réception. Active TI pour une fin de transmission ou RI pour une fin de réception. |

Si deux interruptions de même niveau de priorité sont reçues simultanément, elles seront servies dans l'ordre suivant :
1 : Interruption externe INT0.
2 : Débordement timer 0.
3 : Interruption externe INT1.
4 : Débordement timer 1.
5 : Port série.

Quand le processeur reconnaît une interruption, il positionne un bistable indiquant le niveau de priorité actif. Ensuite, il exécute un « call hardware » et remet à zéro les flags d'interruptions concernés (exceptés INT0, INT1 et TI RI) ; le PC est sauvegardé automatiquement dans la pile, le PSW n'est pas sauvegardé.

Le PC est chargé à une valeur dépendante de la source d'interruption :

| Source | Adresse |
|---|---|
| INT0 interruption externe | 0003H |
| Débordement timer 0 | 000BH |
| INT1 Interruption externe | 0013H |
| Débordement timer 1 | 001BH |
| Port série Emission ou Réception | 0023H |

L'instruction de retour d'une interruption est : RETI.

RETI remet à zéro le bistable de niveau de priorité en cours et recharge le PC avec la valeur qui avait été sauvegardée dans la pile lors de l'exécution de « call hardware ».

L'exécution du programme interrompu reprend là où il avait été quitté.

## 14.1 Logique d'interruption externe

Les interruption externes peuvent être programmées pour être actives sur un niveau ou sur une transition en positionnant les bits IT0 ou IT1 du SFR TCON.

Si ITx = 1, une interruption externe, $\overline{INTx}$, sera active sur un niveau bas (0 volt) sur la broche $\overline{INTx}$. Si ITX = 1, $\overline{INTx}$ sera active sur un front de descente sur $\overline{INTx}$. Le flag d'interruption IEx du SFR TCON est mis à 1 et une demande d'interruption est envoyée au CPU.

Si une interruption est programmée active sur une transition, chaque niveau haut et bas doit être maintenu au moins pendant douze périodes de l'oscillateur pour s'assurer que la transition soit reconnue par la CPU et que le flag d'interruption IEx soit positionné.

IEx sera automatiquement remis à zéro par le CPU à l'entrée de la routine d'interruption.

Si une interruption est programmée active sur un niveau (niveau bas), le niveau bas doit être maintenu au moins pendant douze périodes de l'oscillateur. Cette interruption doit être désactivée avant le retour du service d'interruption sinon une nouvelle interruption sera générée.

$\overline{INT0}$ et $\overline{INT1}$ sont mémorisées dans un registre interne au temps S5P2 et leurs valeurs sont analysées en séquence au cycle machine suivant : le temps de réponse ne sera jamais inférieur à 3 cycles et jamais supérieur à 8 cycles.

## 14.2 Registres de contrôle des interruptions

| Source | Flag | Adresse |
|---|---|---|
| External interrupt 0 | $\overline{INT0}$ si IT0=0<br>IE0 si IT0=1 | P3.2<br>TCON.1 |
| Timer 0 overflow | TF0 | TCON.5 |
| External interrupt 1 | $\overline{INT1}$ si IT1=0<br>IE1 si IT1=1 | P3.3<br>TCON.3 |
| Timer 1 overflow | TF1 | TCON.7 |
| Serial port | TI transmission<br>RI réception | SCON.1<br>SCON.2 |

Reset remet à zéro tous les flags ainsi que IT0 et IT1.

Tous les flags d'interruption peuvent être mis à 1 ou à 0 par programme.

*a) IP registre de priorité des interruptions*

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | X | X | X | PS | PT1 | PX1 | PT0 | PX0 |

- PS : définit le niveau de priorité du port série.
- PS = 1 : niveau de priorité haut du port série.
- PT1 : définit le niveau de priorité du timer 1.
  PT1 = 1 : niveau de priorité haut du timer 1.
- PX1 : définit le niveau de priorité de INT1.
  PX1 = 1 : niveau de priorité haut pour INT1.
- PT0 : définit le niveau de priorité du timer 0.
- PT0 = 1 : niveau de priorité haut du timer 0.
- PX0 : définit le niveau de priorité de INT0.
- PX1 = 1 : niveau de priorité haut pour INT0.

*b) IE registre de validation des interruptions*

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EA | X | X | ES | ET1 | EX1 | ET0 | EX0 |

- EA : Validation de toutes les interruptions.
Si EA = 0, aucune interruption ne sera reconnue.
Si EA = 1, chaque interruption est validée ou invalidée selon son propre bit de validation.
- ES : Validation de l'interruption du port série.
Si ES = 1, l'interruption du port série est validée.
- ET1 : Validation de l'interruption du timer 1.
Si ET1 = 1, l'interruption du timer 1 est validée.
- EX1 : Validation de l'interruption externe $\overline{INT1}$.
Si EX1 = 1, l'interruption externe $\overline{INT1}$ est validée.
- ET0 : Validation de l'interruption du timer 0.
Si ET0 = 1, l'interruption du timer 0 est validée.
- EX0 : Validation de l'interruption externe $\overline{INT0}$.
Si EX0 = 1, l'interruption externe $\overline{INT0}$ est validée.

Tous les bits de IE sont contrôlables par programme et sont remis à zéro par Reset.

## 15 Application d'un 8751 à la programmation de deux PLL (voir chapitre 8 page 235)

Dans cette application (schéma de la figure 4.14), un microcontrôleur est chargé de la programmation de deux synthétiseurs de fréquences via un bus 3 lignes : clock, data, enable.

Pour des raisons de clarté, les sous-ensembles HF des PLL ont été éliminés du schéma pour ne conserver que la circuiterie logique afférente au microcontrôleur.

Ce système est destiné à la programmation d'un émetteur/récepteur. Le premier PLL, U1, est destiné à la stabilisation de la fréquence d'émission $F_E$ ; le second à la fréquence de l'oscillateur local en réception $F_{ol}$.

U1 stabilise la fréquence $F_E$ et U2 stabilise la fréquence $F_{ol} = F_E + F_I$.

$F_I$ est la fréquence intermédiaire à 140 MHz.

La fréquence de fonctionnement de l'émetteur couvre la plage 300 MHz à 427,5 MHz, par pas de 0,5 MHz.

Un oscillateur stable à 10 MHz, U5, délivre un signal au microcontrôleur et aux entrées référence des PLL, U1 et U2.

Pour chacun des PLL, la fréquence stabilisée est égale à :

$$f_{VCO} = (NP + A)f_{REF} / R.$$

$f_{REF}$ est la fréquence de l'oscillateur à quartz à 10 MHz ; N, P, A et R sont des diviseurs programmables internes qui seront chargés par le microcontrôleur U4.

La valeur lue sur le port P2-S1 fixe la fréquence. Soit $x$ cette valeur :

$F_E = 300 + 0{,}5\ x$ MHz pour l'émetteur,

$F_{OL} = F_E + F_I$ pour le récepteur.

La procédure d'introduction des données suit le diagramme des temps de la figure 4.15.

Les diviseurs N, P, A et R sont introduits conformément à la procédure de la figure 4.16.

La détection de verrouillage – broches 7 des PLL, U1 et U2 –, est exploitée par le microcontrôleur.

En cas de défaut, les sorties T0 et T1 rendent compte du défaut sur l'oscillateur de l'émetteur ou du récepteur grâce aux diodes électroluminescentes DL1 et DL2.

La première tâche du concepteur s'achève avec l'établissement du schéma de principe comme celui de la figure 4.14.

Il s'agit ensuite d'écrire et valider le programme.

Il existe deux procédures se découpant chacune en plusieurs étapes : procédure avec ou sans émulateur. L'émulateur est une interface spécifique qui permet au micro-ordinateur de remplacer (émuler) le microcontrôleur ou le microprocesseur.

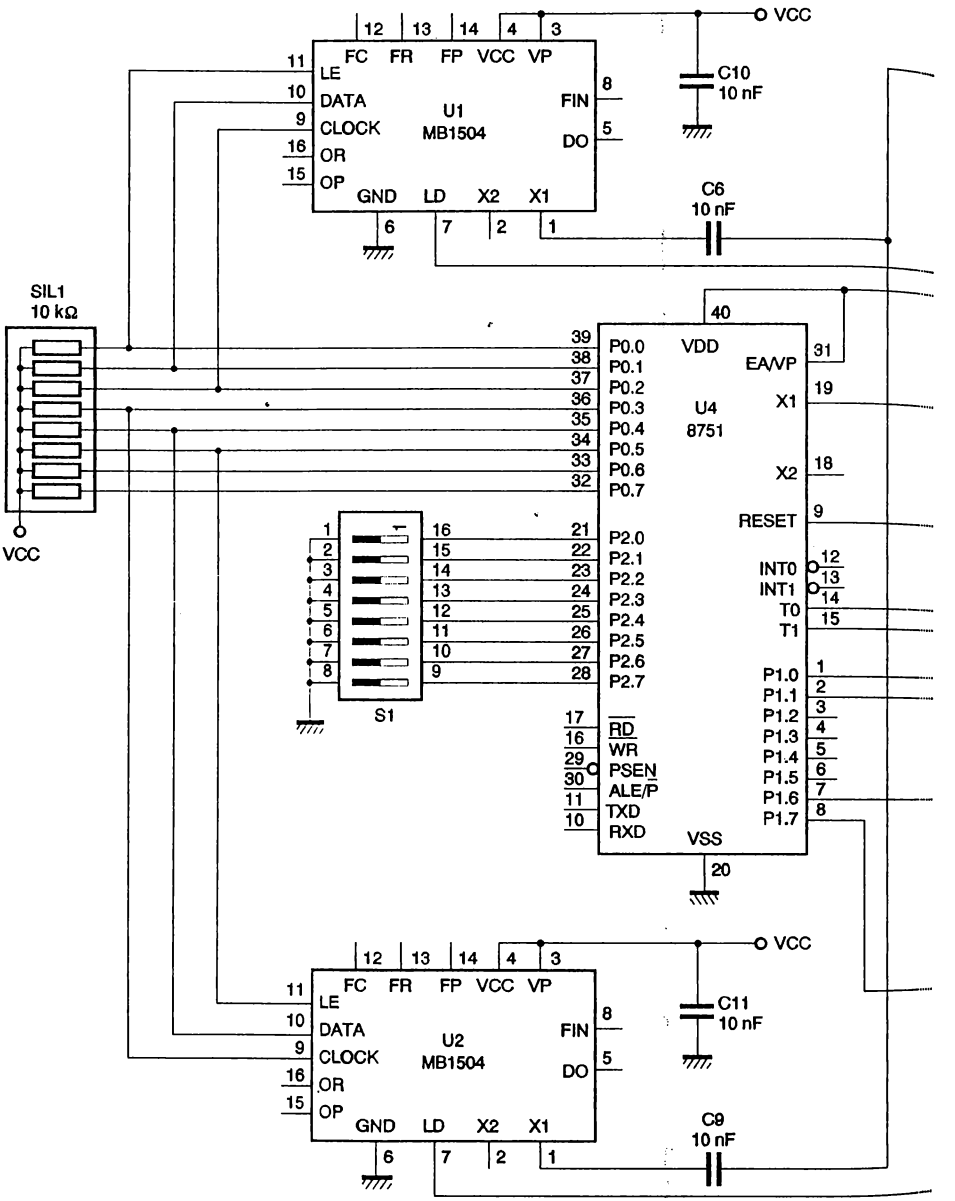



Fig. 4.14
Schéma d'application d'un 8751 à la programmation de deux PLL.

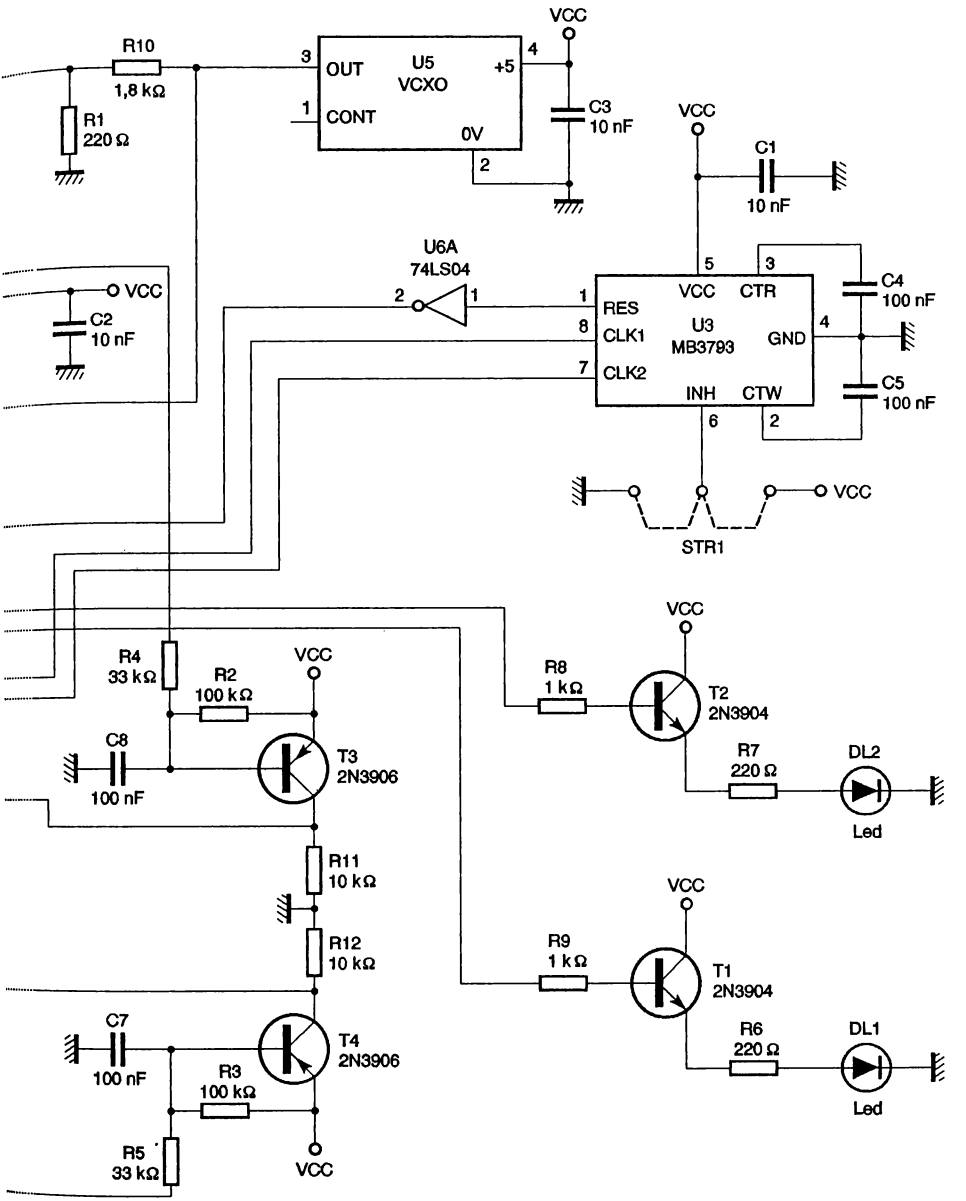



Fig. 4.14 (suite)

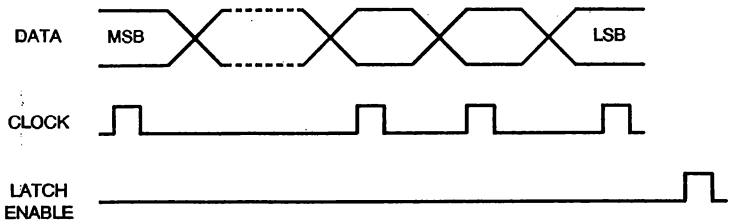



Fig. 4.15
Introduction des données dans les PLL MB 1504.

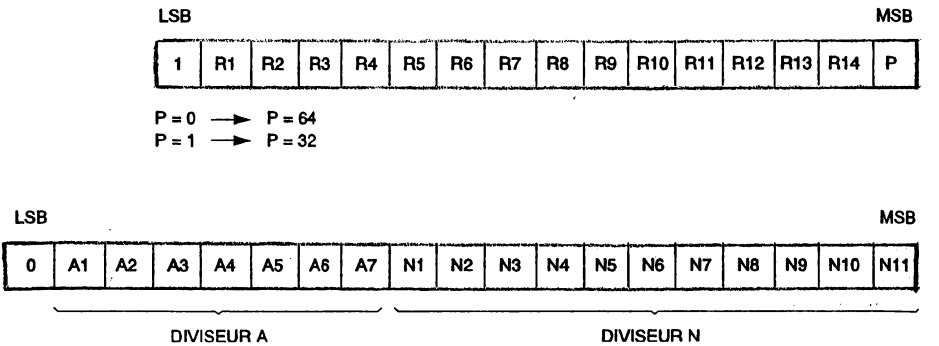

LSB
MSB
1 R1 R2 R3 R4 R5 R6 R7 R8 R9 R10 R11 R12 R13 R14 P
P = 0 → P = 64
P = 1 → P = 32
LSB
MSB
0 A1 A2 A3 A4 A5 A6 A7 N1 N2 N3 N4 N5 N6 N7 N8 N9 N10 N11
DIVISEUR A
DIVISEUR N

Fig. 4.16
Procédure d'introduction de la valeur des diviseurs.

### *15.1 Procédure sans émulateur*

La première tâche du concepteur programmeur est l'écriture du programme. Pour cette première phase, seul un logiciel de traitement de texte ou éditeur est nécessaire.
La deuxième phase consiste à compiler le programme. La compilation peut générer un certain nombre de fichiers comme des fichiers assembleurs ou des fichiers de code binaire directement compréhensibles par le microcontrôleur ou le microprocesseur.

La phase finale consiste à graver le code dans une mémoire qui sera connectée au montage final.

Les tests fonctionnels sont ensuite nécessaires pour valider le programme. En cas de dysfonctionnement, la procédure complète doit être appliquée de nouveau.

Cette solution est la plus économique en terme de matériel – absence d'émulateur – mais la plus longue en temps de développement.

Il est en général assez difficile de détecter les erreurs par cette procédure constituée d'une suite d'essais - erreurs.

Dans des cas simples, elle peut malgré tout donner satisfaction.

### *15.2 Procédure avec émulateur*

Cette procédure est identique à la précédente jusqu'à la compilation du programme.

À ce stade, l'émulateur est connecté à la cible – montage d'essai – et un micro-ordinateur émule le programme.

Il est alors facile d'effectuer toutes les séries de test en surveillant le programme – contenu des registres par exemple –.

Le programme peut aussi être déroulé en mode pas à pas.

En cas de dysfonctionnement, les erreurs sont localisées beaucoup plus facilement que dans la situation précédente.

Les phases répétées d'écriture – effacement des mémoires EEPROM ou microcontrôleur UV – sont évitées.

Le gain est double : gain de temps et gain économique car les microcontrôleurs UV supportent assez mal les cycles écriture – effacement répétés.

### *15.3 Programme*

Les lignes qui suivent donnent, à titre d'exemple, le programme écrit en C51 pour l'exemple de la figure 4.14.

```
C51 COMPILER V3.40,  51FBER

DOS C51 COMPILER V3.40, COMPILATION OF MODULE 51FBER
OBJECT MODULE PLACED IN D:\C51P\TRAV\51FBER.OBJ
COMPILER      INVOKED      BY:      C:\C51P\BIN\C51.EXE
D:\C51P\TRAV\51FBER.C

stmt level    source

1
/*************************************************/
2       /*        PROG   80C51FB  LE 30/11/94        */
3
/*************************************************/
4
5       #pragma small code debug oe
6
7       #include <reg51f.h>
8       #include <stdio.h>
9       #include <intrins.h>
10
11        extern void _nop_(void);
12        void lec();
13        void calc1();
14        void serial1();
```

```
15        void calc2();
16        void serial2();
17        void alarme();
18        /*********** déclaration des variables **********/
19
20        sbit P00 =P0^0 ;    /* variables pour l'émetteur */
21        sbit P01 =P0^1 ;
22        sbit P02 =P0^2 ;
23
24        sbit P03 =P0^3 ;    /* variables pour le récepteur */
25        sbit P04 =P0^4 ;
26        sbit P05 =P0^5 ;
27
28        sbit P16 =P1^6 ;    /* entrée info émetteur  */
29        sbit P17 =P1^7 ;    /* entrée info récepteur */
30
31        sbit P34 =P3^4 ;    /* sortie signalisation émetteur */
32        sbit P35 =P3^5 ;    /* sortie signalisation récepteur */
33
34        char i ;
35        unsigned char s1,p1,al1;
36        unsigned int r1,N1,A1;
37        unsigned char s2,p2,al2;
38        unsigned int r2,N2,A2;
39        unsigned long fe ,fr;
40
41        /*************** Prog Principal ***************/
42        main()
43          {
44  1       lec();
45  1       calc1();
46  1       serial1();
47  1       calc2();
48  1       serial2();
49  1       while(1)
50  1        alarme() ;
51  1       }
52        /*************** lecture de S1 ***************/
53        void lec()
54          {
55  1       s1=P2;
56  1       }
57        /******** calcul de A et N  pour l'émetteur ********/
58        void calc1()
59          {
60  1       r1=120;              /* fxtal = 12 Mhz */
61  1       p1=32;              /* fe = (((N1 * p1) + A1)* fxtal)/r1 */
62  1       fe = 3000+5*s1 ;
```

```
63  1      N1=fe/p1;
64  1      A1=fe%p1;
65  1     }
66       /***** sérialisation des données pour l'émetteur *****/
67        void serial1()
68         {
69  1      P02 = 0 ;              /* LE = 0 */
70  1      P01 = 1 ;              /* 0-> SW=64   1-> SW=32 */
71  1      _nop_() ;              /* temps mort de 1us */
72  1      P00 = 1 ;              /* impulsion clk pour validation */
73  1      P00 = 0 ;
74  1      for(i=13 ; i>-1 ;i--)
75  1       {
76  2      P01 = (r1 >> i)  & 1 ; /* sortie  data MSB en premier */
77  2        _nop_() ;            /* temps mort de 1us */
78  2        P00 = 1 ;            /* impulsion clk pour validation */
79  2        P00 = 0 ;
80  2       }
81  1      P01 = 1 ;              /* bit de contrôle ... 1 */
82  1      _nop_() ;
83  1      P00 = 1 ;
84  1      P00 = 0 ;
85  1      _nop_() ;
86  1      P02 = 1 ;              /* impulsion de validation  LE */
87  1      P02 = 0 ;
88  1
89  1      for(i=10 ; i>-1 ;i--)
90  1       {
91  2      P01 = (N1 >> i) & 1 ;  /* sortie data MSB en premier */
92  2        _nop_() ;
93  2        P00 = 1 ;            /* impulsion clk pour validation */
94  2        P00 = 0 ;
95  2       }
96  1      for(i=6 ; i>-1 ;i--)
97  1       {
98  2      P01 = (A1 >> i) & 1 ;  /* sortie data MSB en premier */
99  2        _nop_() ;
100 2        P00 = 1 ;            /* impulsion clk pour validation */
101 2        P00 = 0 ;
102 2       }
103 1      P01 = 0 ;               /* bit de contrôle ... 1 */
104 1      _nop_() ;
105 1      P00 = 1 ;
106 1      P00 = 0 ;
107 1      _nop_() ;
108 1      P02 = 1 ;              /* impulsion de validation  LE */
109 1      P02 = 0 ;
110 1      alarme();
```

```
111  1     }
112       /******* calcul de A et N  pour le récepteur *******/
113       void calc2()
114        {
115  1      r2=120;            /* fxtal = 12 Mhz */
116  1      p2=32;            /* fr = (((N2 * p2)+ A2)* fxtal)/r2 */
117  1      fr = 4400+5*s1 ;
118  1      N2=fr/p2;
119  1      A2=fr%p2;
120  1      }
121     /**** sérialisation des données pour le Récepteur ****/
122       void serial2()
123        {
124  1      P05 = 0 ;              /* LE = 0 */
125  1      P04 = 1 ;            /* 0-> SW=64   1-> SW=32 */
126  1      _nop_() ;             /* temps mort de 1us */
127  1      P03 = 1 ;            /* impulsion clk pour validation */
128  1      P03 = 0 ;
129  1      for(i=13 ; i>-1 ;i--)
130  1       {
131  2     P04 = (r2 >> i)  & 1 ; /* sortie data MSB en premier */
132  2        _nop_() ;           /* temps mort de 1us */
133  2       P03 = 1 ;          /* impulsion clk pour validation */
134  2        P03 = 0 ;
135  2       }
136  1      P04 = 1 ;             /* bit de contrôle ... 1 */
137  1      _nop_() ;
138  1      P03 = 1 ;
139  1      P03 = 0 ;
140  1      _nop_() ;
141  1      P05 = 1 ;             /* impulsion de validation  LE */
142  1      P05 = 0 ;
143  1
144  1      for(i=10 ; i>-1 ;i--)
145  1       {
146  2    P04 = (N2 >> i) & 1 ;  /* sortie data MSB en premier */
147  2        _nop_() ;
148  2       P03 = 1 ;          /* impulsion clk pour validation */
149  2        P03 = 0 ;
150  2       }
151  1      for(i=6 ; i>-1 ;i--)
152  1       {
153  2    P04 = (A2 >> i) & 1 ;  /* sortie data MSB en premier */
154  2        _nop_() ;
155  2       P03 = 1 ;          /* impulsion clk pour validation */
156  2        P03 = 0 ;
157  2       }
158  1      P04 = 0 ;             /* bit de contrôle ... 1 */
```

```
159  1      _nop_() ;
160  1      P03 = 1 ;
161  1      P03 = 0 ;
162  1      _nop_() ;
163  1      P05 = 1 ;            /* impulsion de validation LE */
164  1      P05 = 0 ;
165  1      alarme();
166  1     }
167          /********** signalisation émetteur **********/
168          void alarme()
169           {
170  1        al1=P16 ;
171  1        if (al1 == 1)
172  1          P34 = 1 ;
173  1        else
174  1          P34 = 0 ;
175  1        al2=P17;
176  1        if (al2 == 1)
177  1          P35 = 1 ;
178  1        else
179  1          P35 = 0 ;
180  1       }
181          /****************** FIN  ******************/
```

## Exemple d'un microprocesseur

Pour l'exemple d'un microprocesseur, la famille 8086 a été retenue pour une raison évidente : son utilisation dans les micro-ordinateurs PC, qui feront l'objet d'une description sommaire à la fin de ce chapitre.

Dans cette famille, le premier processeur 16 bits est le 8086 qui a eu de nombreux successeurs jusqu'au Pentium 32 bits.

Nous nous intéresserons à une version intermédiaire, le 80286.

Le brochage du circuit intégré est représenté à la figure 4.17. La largeur du bus de données est de 16 bits. Le bus d'adresse, constitué de 24 bits, peut gérer un espace physique de 16 Mo.

Le processeur 80286 peut fonctionner selon deux modes :
– en mode réel qui présente un système d'adressage compatible avec celui du 8086 mais un jeu d'instructions étendu et sensiblement plus complet ;
– en mode protégé qui intègre des mécanismes complexes comme le contrôle des débordements de zone, la gestion des tâches et commutation de contexte, la protection des données à plusieurs niveaux de privilège d'exécution en fonction de leur niveau de privilège propre.

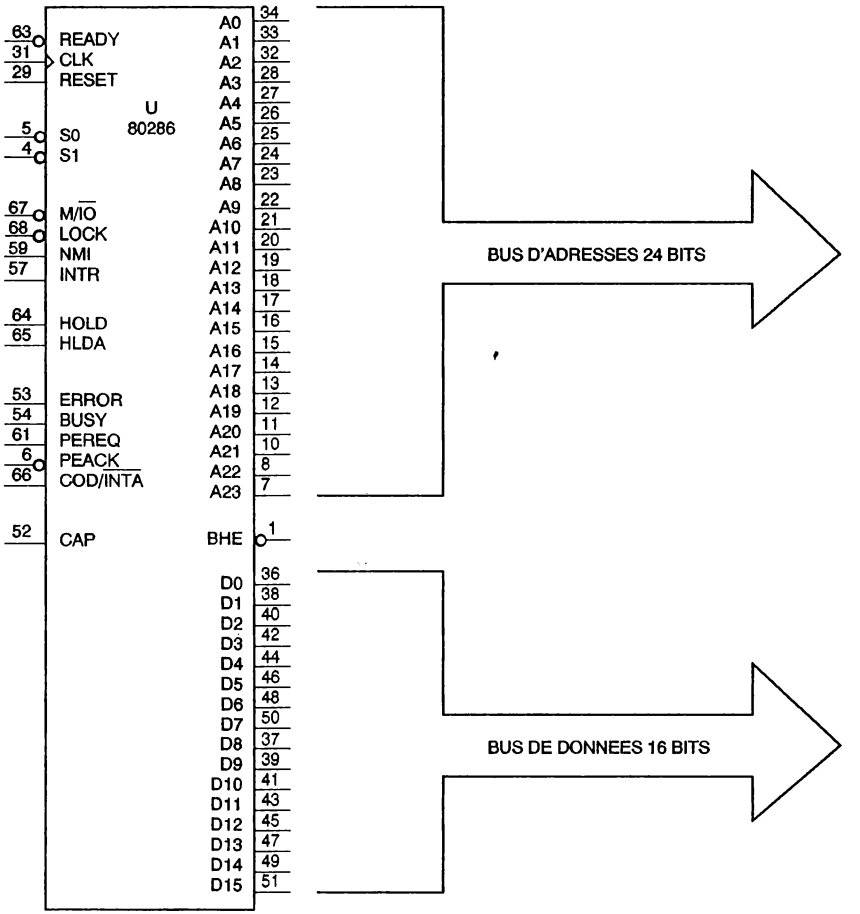



Fig. 4.17
Microprocesseur 80286.

Contrairement à un microcontrôleur, un microprocesseur ne peut fonctionner seul. Quelle que soit l'application (machine universelle comme le PC ou machine dédiée), il est donc accompagné d'un certain nombre de circuits périphériques. Dans le cas du 80286, ces circuits périphériques sont :

– Un générateur d'horloge 82284. Les trois fonctions essentielles de ce circuit sont la synchronisation du signal RESET, la génération de l'horloge de base, la synchronisation du signal READY permettant l'accès à un circuit mémoire ou interface d'entrée/sortie.

– Un coprocesseur mathématique optionnel 80287. Ce processeur spécialisé dans les traitements numériques peut généralement accélérer les calculs portant sur des nombres réels flottants ou sur des nombres entiers en format étendu dans des rapports voisins de 100, en général fonctions de l'opération.

L'intérêt majeur de ce coprocesseur réside dans sa facilité de connexion au processeur maître car son bus est directement branché en parallèle sur celui du 80286.

Le 80287 reconnaît automatiquement les opérations pour lesquels il est concerné.

La présence ou l'absence de ce coprocesseur reste transparente au programmeur jusqu'au moment de l'édition des liens. La plupart des compilateurs des langages évolués comportent une option avec ou sans coprocesseur.

Les circuits annexes sont :
– un contrôleur de RAM dynamique 8207,
– des tampons bidirectionnels 8286-8287,
– des bascules 8283-8282,
– un contrôleur d'interruption 8259A.

## Architecture système

Une architecture autour d'un 80286 comporte deux types de bus comme le montre le synoptique de la figure 4.18 :
– un bus local composé des signaux directement produits par le processeur lui-même (adresses données et signaux de contrôle),
– un bus système qui relie ou connecte le processeur à des ressources publiques de mémoire ou d'entrée/sortie.

Cette architecture est utilisée dans les systèmes Intel Multibus I et Multibus II.

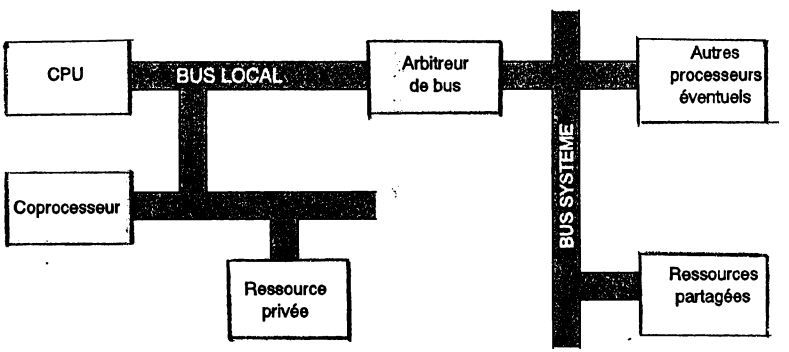



Fig. 4.18
Architecture système multiprocesseur autour d'un 80286.

Une architecture plus simple est représentée à la figure 4.19.

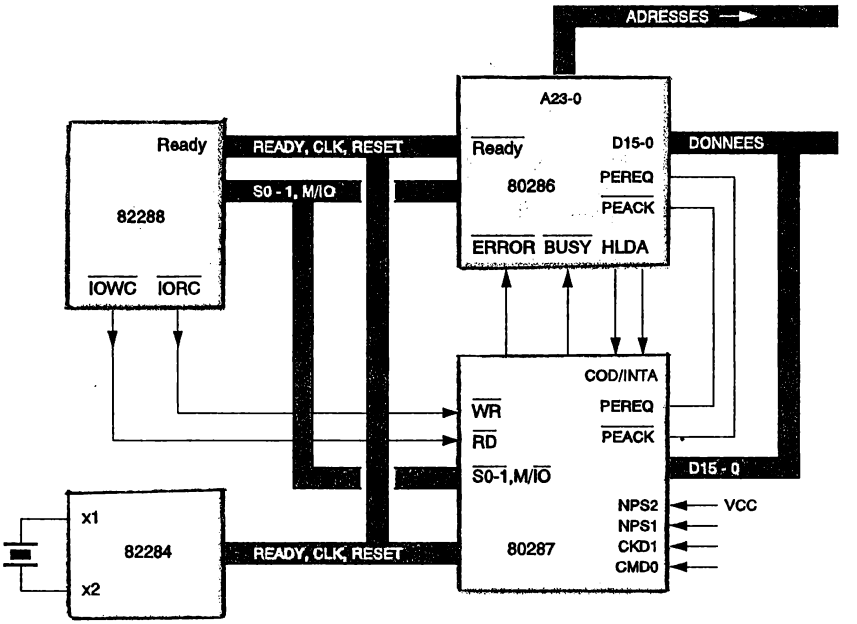



Fig. 4.19
Architecture du 80286 avec ses circuits périphériques.

## 2 Structure interne du processeur 80286

Dérivée du 8086, l'unité centrale 80286 a été conçue comme un microprocesseur 16 bits autorisant une utilisation dans un environnement multitâches - multi-utilisateurs.

Cette caractéristique n'était pas employée par Microsoft dans son système d'exploitation DOS.

Le boîtier intègre de façon câblée un certain nombre de mécanismes de base permettant l'implantation simple d'un système d'exploitation performant.

Les principaux mécanismes implantés traitent le partage de la mémoire, la protection de celle-ci par niveaux de privilège répartis en anneau, le contrôle de non débordement de zone lors d'un accès mémoire par une tâche spécifique.

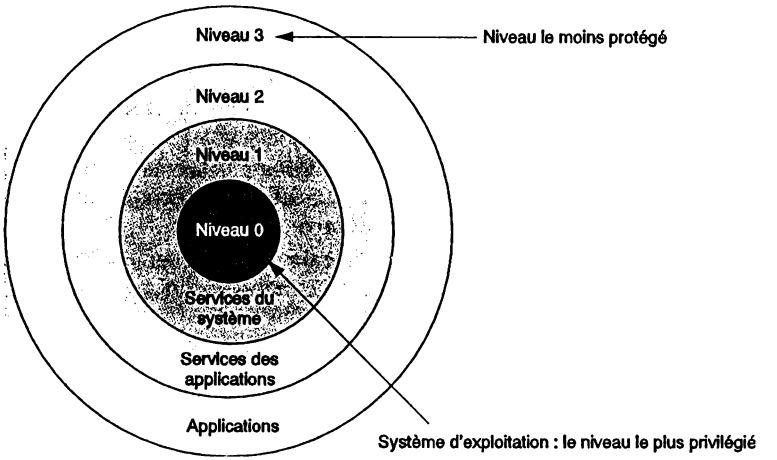



Fig. 4.20
Les quatre niveaux de privilège du 80286.

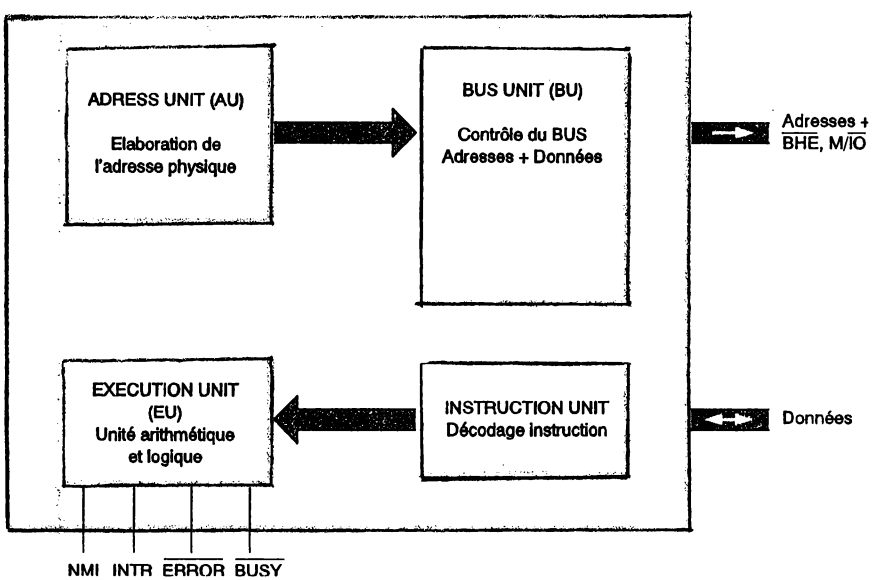



Fig. 4.21
Structure interne du 80286.

Les quatre niveaux de privilège sont schématisés par la figure 4.20. Le schéma de la figure 4.21 représente la structure interne du 80286 constitué de quatre sous-ensembles élémentaires, détaillés ci-après.

### 2.1 Le bus unit

Le bus unit réalise toutes les opérations portant sur les bus d'entrée/sortie mémoire ou périphériques : génération des adresses, des données, des signaux de commande nécessaires aux accès mémoire.

Le bus unit établit un régime pipeline en accès au bus, par la lecture anticipée des instructions qui sont ensuite mémorisées à l'intérieur de l'instruction unit.

Cette lecture anticipée permet de stocker jusqu'à six octets d'instruction ; ce processus permet une plus grande vitesse de traitement.

### 2.2 L'instruction unit

L'instruction unit reçoit les instructions de la file des six octets du bus unit, décode ces instructions et les stocke, décodées, dans une file d'attente pouvant contenir jusqu'à trois instructions complètes.

### 2.3 L'exécution unit

Les instructions décodées sont alors récupérées par l'exécution unit qui les exécute et peut éventuellement solliciter le bus unit pour réaliser des accès en lecture ou en écriture sur le bus de données.

### 2.4 L'adress unit

La fonction de cette unité est la gestion de la mémoire et la génération des adresses physiques utilisées par le bus unit.

La génération des adresses nécessite la présence d'une unité arithmétique de 24 bits pour prétendre à l'accès effectif sur une zone de 16 Mo.

La gestion de la mémoire comporte également les contrôles de protection.

## 3 Segmentation de la mémoire

### 3.1 Mode réel

Le microprocesseur 80286 peut adresser 1 Mo de mémoire RAM en mode réel et 16 Mo en mode protégé.

Dans les deux cas, la mémoire globale est tronçonnée en segments de 64 Ko.

Un segment est donc défini par son adresse physique absolue de départ dans la mémoire et sa longueur maximale de 64 Ko.

En règle générale, un segment devra commencer dans la mémoire à une adresse divisible par 16.

La figure 4.22 montre que les segments peuvent être consécutifs, disjoints, se chevaucher ou se superposer.

À un instant donné, l'adresse d'une information est définie à l'aide du contenu des deux registres CS et IP.

Le registre CS est le registre sélecteur de segment et le registre IP donne la valeur du déplacement dans ce segment. Ce déplacement est aussi appelé offset.

L'adresse complète est notée CS:IP.

Les deux registres CS et IP sont des registres 16 bits. L'adresse réelle sur 20 bits peut se calculer par la relation 16.CS + IP qui donne un résultat sur 20 bits.

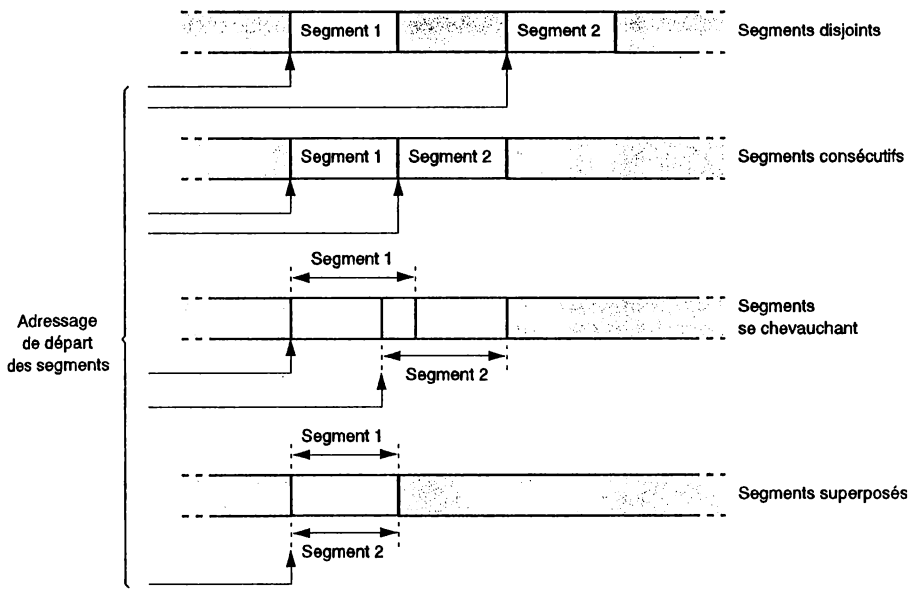



Fig. 4.22
Répartition des segments dans la mémoire.

### 3.2 Mode protégé

Dans cette configuration, le 80286 utilise à plein les 24 bits de son espace d'adressage, ce qui autorise l'accès à 16 Mo de mémoire. Une mémoire cache, interne à l'Adress Unit, permet alors d'effectuer la traduction d'adresse, d'évaluer les niveaux de protection et d'élaborer éventuellement cette adresse physique de la mémoire cible. On parle ici de mémoire virtuelle car l'espace adressable est beaucoup plus large que l'espace physique théorique. On suppose donc que des parties de la mémoire sont supportées par des mémoires auxiliaires et chargées lors de leur utilisation effective.

En mode protégé, comme en mode réel, l'adresse peut être exprimée par CS:IP.

### *3.3 Sélection du segment*

En mode protégé, le registre de segment contient toujours un mot sur 16 bits chargé par le programme mais ce mot ne pointe plus directement un segment : afin d'accroître l'espace couvert, il pointe une entrée dans une **table de description de segments** ; c'est elle qui indique le bon segment. Il s'agit, en quelque sorte, d'un adressage indirect du segment.

Le registre de segment concerné s'appelle désormais un **sélecteur de segment.** Il contient un mot de 16 bits organisés en 13 + 1 bits d'adresse (un index sur 13 bits plus un bit de sélection), plus 2 bits de niveaux de privilège. Ces deux derniers bits sont référencés **RPL**, pour « Requested Privilege Level ».

## 4 Les registres du 80286

La figure 4.23 regroupe les registres équipant le processeur 80286. Ils sont répartis en trois groupes :
- les registres généraux,
- les registres de segmentation,
- les registres d'état et de contrôle.

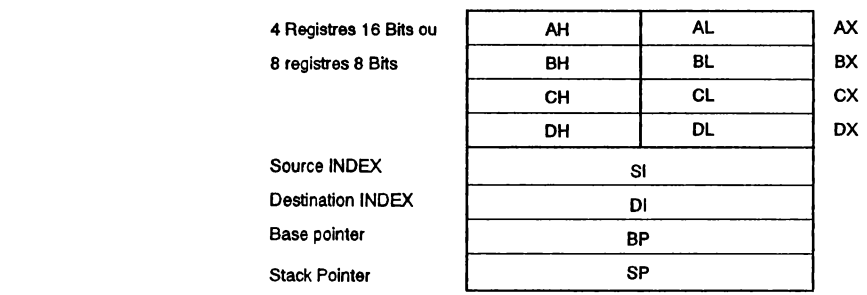



Fig. 4.23
Registres internes du 80286.

### *4.1 Registres généraux*

Il s'agit de 8 registres 16 bits. Le 80286 n'a pas de registre accumulateur. Chaque registre peut être employé pour servir d'opérande ou de résultat dans une opération arithmétique ou logique.

Certains de ces registres sont parfois dédiés à des opérations particulières.

– AL, AX : sont obligatoirement utilisés comme supports d'opérande dans les instructions d'entrées/sorties ainsi que dans les multiplications et divisions.

– CX : compteur de boucle dans les instructions de comptage (Loop) ou dans les instructions de traitement de chaîne ainsi que dans les décalages de plusieurs pas.

– DX : support d'adresse dans les instructions d'entrées/sorties ou poids forts d'une quantité 32 bits pour la multiplication ou la division.

– SP : pointeur de pile.

– BX, BP : sont très fréquemment utilisés dans les modes d'adressage indirect et sont de ce fait appelés registres de base. BP est plus spécifiquement dédié à la pile et trouve son utilisation principale dans la récupération des paramètres envoyés à une procédure par un programme appelant.

– SI, DI : s'ils peuvent être utilisés pour l'adressage indirect, ils sont également dédiés à la fonction d'index dans l'adressage indirect indexé ou dans les instructions de traitement de chaînes. Ils sont de fait appelés registres d'index.

### *4.2 Registres de segmentation*

Le 80286 est muni de quatre registres de segments distincts (DS, ES, SS, CS) permettant donc d'accéder immédiatement à des objets situés dans quatre segments logiques différents, situés à une implantation quelconque en mémoire.

Le code segment (CS) pointe le segment contenant le programme en cours d'exécution.

Le stack segment (SS) pointe le segment dans lequel se trouve utilisée la zone de pile courante, indexée par le Stack Pointer SP ou le Base Pointer BP.

Le data segment (DS) référence le segment des données utilisées par le programme en cours.

L'extra segment (ES) est analogue au registre DS ; il permet éventuellement d'accéder à un deuxième segment de données.

### *4.3 Registres d'état et de contrôle*

Le 80286 contient deux registres d'état et de contrôle. Le premier registre contient les 11 flags – indicateurs – dont la signification est donnée à la figure 4.24. Le tableau de la figure 4.25 détaille la signification de chaque bit.

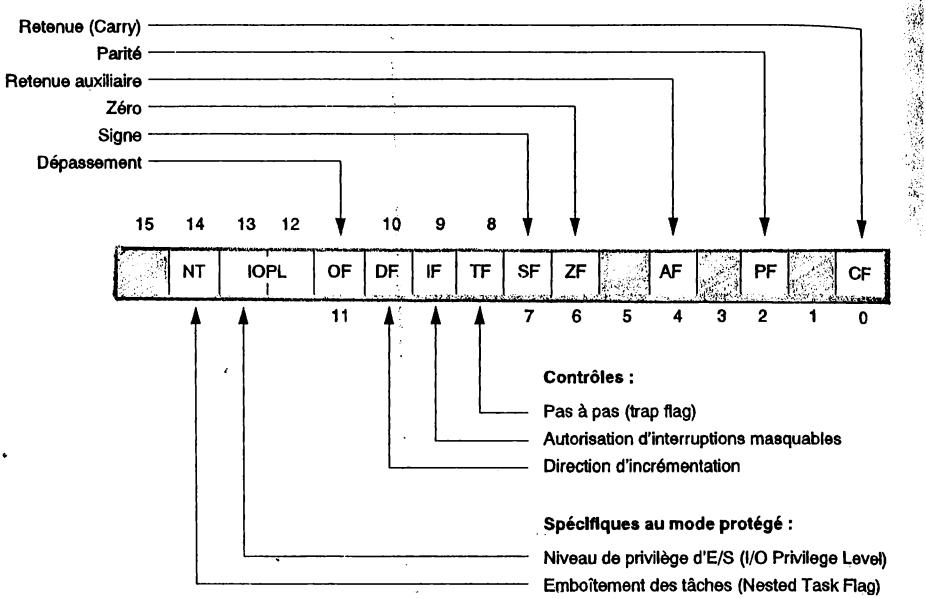

Retenue (Carry)
Parité
Retenue auxiliaire
Zéro
Signe
Dépassement
15 14 13 12 10 9 8
NT IOPL OF DF IF TF SF ZF AF PF CF
11 7 6 5 4 3 2 1 0
Contrôles :
Pas à pas (trap flag)
Autorisation d'interruptions masquables
Direction d'incrémentation
Spécifiques au mode protégé :
Niveau de privilège d'E/S (I/O Privilege Level)
Emboîtement des tâches (Nested Task Flag)

Fig. 4.24
Registres contenant les *flags*.

| Position du bit | Parfois marqué | Nom | Fonction |
|---|---|---|---|
| 0 | C | CF | Carry - Mis à 1 sur retenue, à 0 autrement |
| 2 | P | PF | Parité - A 1 si l'octet de faible poids comporte un nombre pair de 1 |
| 4 | A | AF | Retenue sur quartet de faible poids du registre AL |
| 6 | Z | ZF | Zéro : à 1 sur un résultat nul |
| 7 | S | SF | Signe : recopie le bit de fort poids avec 0 = positif, et 1 = négatif |
| 8 | T | TF | Trap (pas à pas) : à 1, déclenche une interruption après la prochaine instruction, puis revient à 0 |
| 9 | I | IF | Autorisation des interruptions masquables |
| 10 | D | DF | Direction d'incrémentation. A 1, décrémente l'index ; à 0, l'incrémente |
| 11 | O | OF | Dépassement sur résultat signé ; passe alors à 1 |
| 12 et 13 | | IOPL | Niveau de privilège. N'intervient qu'en mode protégé |
| 14 | | NT | Emboîtement des tâches. N'intervient qu'en mode protégé |

Fig. 4.25
Signification des *flags*.

Le deuxième registre témoigne de l'état de la machine, il est baptisé Machine Status Word MSW (voir figure 4.26).

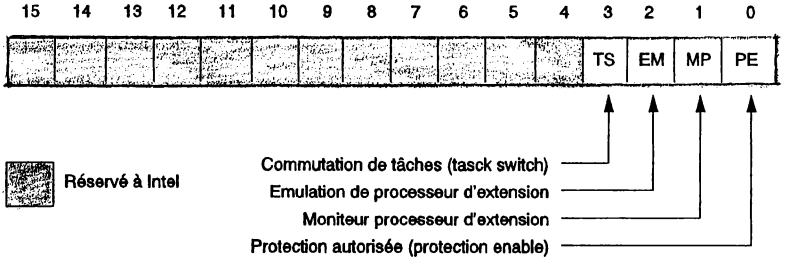

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
TS EM MP PE
Réservé à Intel
Commutation de tâches (tasck switch)
Emulation de processeur d'extension
Moniteur processeur d'extension
Protection autorisée (protection enable)

Fig. 4.26
Registre d'état de la machine.

## 5 Quelques instructions assembleur de la famille 8086

La liste des instructions suivantes est destinée à donner une idée du langage assembleur de la famille 8086.

Seules quelques instructions simples ont été sélectionnées.

### *5.1 Instructions arithmétiques*

La destination et la source A et B peuvent être un registre ou une mémoire, la source peut être une donnée immédiate.

- MOV : destination, source.
- ADD : destination, source.
- ADD B, A : le résultat A + B est mis dans B, et A est inchangé.
- SUB B, A : le résultat B − A est mis dans B, et A est inchangé.
- AND B, A : le résultat est mis dans B, et A est inchangé.
- OR B, A.
- CMP B, A : compare A et B et positionne les flags.
- INC RM : RM est remplacé par RM + 1.
- DEC RM : RM est remplacé par RM − 1.
- NOT RM : donne le complément de RM.
- NEG RM : donne le complément à 2 de RM.

### *5.2 Instructions sur la pile*

- PUSH RM : place RM dans la pile.
- POP RM : extrait RM de la pile.

### *5.3 Instructions de contrôle*

- JMP : branchement inconditionnel.
- J** : branchement conditionnel : égalité, plus grand, plus petit, etc.
- CALL : appel à un sous-programme.
- RET : retour d'un sous-programme.
- IRET : retour d'interruption.
- STI : validation des interruptions.
- CLI : invalidation des interruptions.

### *5.4 Entrées/sorties*

- IN AL, DX : met dans AL la valeur lue sur le port DX .
- OUT DX, AL : positionne la valeur AL dans le port DX.

## Le micro-ordinateur : architecture, caractéristiques et utilisation

L'architecture du micro-ordinateur le plus général est représentée à la figure 4.27. Ce schéma s'applique à l'IBM PC ainsi qu'à d'autres machines comme les stations de travail en général plus performantes. Tous les sous-ensembles sont reliés par l'intermédiaire d'un bus de données.

Les divers constituants seront examinés brièvement dans l'ordre de la figure 4.27, de gauche à droite.

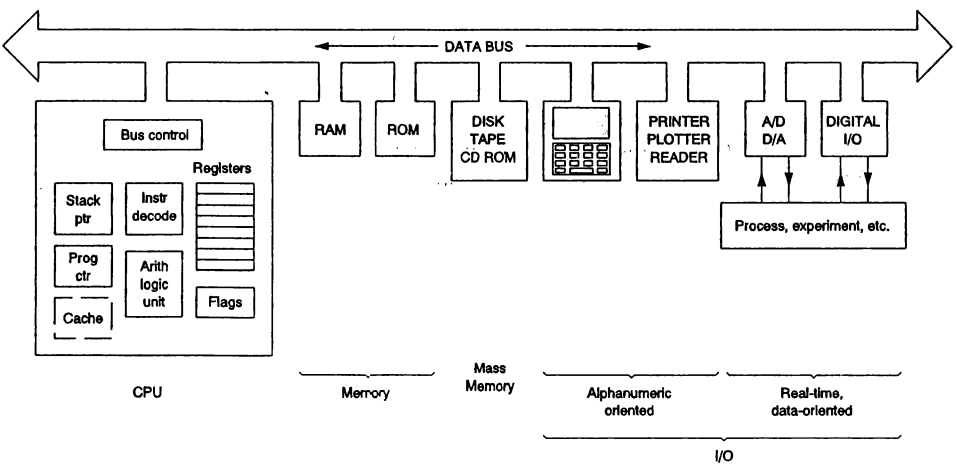



Fig. 4.27
Architecture du micro-ordinateur.

### 1 CPU ou processeur

La CPU est le centre de la machine. Elle est elle-même constituée d'un certain nombre de sous-ensembles qui ont été examinés antérieurement dans la description d'un microcontrôleur et d'un microprocesseur.

### 2 Mémoires

Classiquement, le processeur est relié à la mémoire données RAM et à la mémoire programme ROM.

La plupart des micro-ordinateurs performants sont équipés de 16 Mo de RAM. Ce chiffre peut être comparé avec les 64 Ko qui équipaient les premières machines dotées de 8086/8 (1980).

La mémoire ROM contient les programmes nécessaires au démarrage de la machine et au système d'exploitation. Il faut remarquer que la mémoire ROM ne contient pas les programmes d'utilisation ou les programmes éventuellement écrits par l'utilisateur.

## 3 Mémoire de masse

Contrairement aux machines dédiées à un traitement ou à une application spécifique, les micro-ordinateurs comportent au moins trois types de mémoire de masse :
– les disquettes souples (« floppy disks ») peuvent enregistrer une capacité de données comprise entre quelques centaines de Ko et 2 Mo environ. Les formats les plus répandus sont 360 Ko, 720 Ko, 1.2 Mo et 1.44 Mo ;
– les disques durs (« winchester ») ont des capacités beaucoup plus importantes, dépassant 1 Go (giga-octet) ;
– le CD-ROM est un disque optique similaire au compact disc audio pouvant stocker plus de 600 Mo d'informations.

Dans les cas où les applications sont critiques, le PC peut être muni d'un enregistreur à bande magnétique dont le principal rôle est la sauvegarde des informations présentes dans les autres mémoires de masse : le disque dur.

Le CD-ROM ne permet qu'une et une seule opération d'écriture, ce qui n'est pas le cas des supports magnétiques (disque souple ou disque dur).

Les accès aux mémoires de masse sont beaucoup plus lents que les accès à la mémoire volatile RAM. En général, les programmes sont transférés de la mémoire de masse (disque dur) à la mémoire vive (RAM) où ils sont exécutés.

Si la quantité de mémoire vive installée dans le micro-ordinateur est suffisante, tous les programmes d'usage courant peuvent au démarrage être transférés en RAM. Cette procédure accélère les temps d'exécution en évitant la lenteur relative du disque dur.

Cette procédure doit être accompagnée d'une sauvegarde périodique du travail en cours : transfert du travail de la zone de mémoire RAM vers la zone de mémoire de masse.

## 4 Clavier, écran et souris

Le micro-ordinateur est capable d'effectuer plusieurs millions d'opérations par seconde. Il est essentiel que l'homme et la machine puissent communiquer pour que la puissance de calcul puisse réellement être à la **disposition** de l'utilisateur.

La communication doit s'établir dans les deux sens : de l'homme vers la machine et de la machine vers l'homme.

Le clavier et la souris servent d'interface de l'utilisateur vers la machine. Ces éléments permettent l'introduction des données et déterminent les actions ou calculs à entreprendre.

Le clavier est essentiellement utilisé pour l'introduction de caractères alphanumériques, par exemple pour les traitements de texte ou la programmation et la souris est employée dans les applications graphiques : dessins, schémas, etc.

L'écran est l'interface de la machine vers l'utilisateur. Les premiers écrans, monochromes, ne permettaient que l'affichage de 25 lignes de 40 ou 80 caractères.

Très rapidement les écrans monochromes ont été remplacés par des écrans couleur. Les standards d'affichage les plus usuels sont dans l'ordre chronologique :
- CGA : 320 × 200, 4 couleurs,
- EGA : 640 × 350, 4 ou 16 couleurs,
- VGA : 640 × 480, 16 couleurs,
- Super VGA : 800 × 600, 16 couleurs.

Des résolutions de 1024 × 768 voire 1280 × 1024 avec 16 millions de couleurs sont désormais standards. Ceci signifie simplement que chaque composante fondamentale, (rouge, verte et bleue) est numérisée et stockée sur un octet. À une page écran correspond donc un plan mémoire, et plus la définition est élevée, plus le plan mémoire est important.

Pour obtenir la meilleure lisibilité et des caractères ou des graphiques, on cherche à augmenter la résolution.

Plus la résolution est importante, plus la quantité d'informations par image est importante.

Pour les applications graphiques, il est souhaitable de disposer de cartes d'interface graphique à haute résolution, équipées d'une importante quantité de mémoire RAM, associées à un écran de grandes dimensions – 20 pouces –.

Le format de l'écran est en général 4-3 (largeur : 4 et hauteur : 3), la taille de l'écran fait toujours référence à la diagonale.

Si la machine est dédiée à la programmation ou au traitement de texte, des résolutions de type VGA et des écrans de 17 pouces peuvent être suffisants.

Des considérations ergonomiques doivent aussi être introduites pour la sélection de la carte écran et de l'écran associé.

## 5 Imprimantes et traceurs

Les résultats des travaux effectués sur la machine sont archivés dans sa mémoire de masse (disque dur).

La trace, écrite, reste le meilleur moyen de diffuser, même si les machines peuvent être mises en réseau.

Les premières imprimantes, à aiguilles, bruyantes et lentes, étaient juste suffisantes pour l'édition des listings.

La diffusion de documents édités par ce procédé était difficilement envisageable.

À contrario, les imprimantes Laser donnent d'excellents résultats avec des résolutions élevées 600 dpi (« dots per inch ») (points par pouce).

Le format A4 est le plus approprié pour les documents texte. Pour cette raison, la plupart des imprimantes Laser sont seulement capables de gérer ce format.

Les machines haut de gamme sont capables de traiter des documents au format A3, ce qui est un atout intéressant dans le cas de graphiques.

Dans de nombreux cas, les graphiques ont des formats bien supérieurs : A2, A1 et même A0. Bien entendu, le micro-ordinateur est capable d'opérer une réduction. Un schéma au format A1 pourra être réduit pour être tiré sur une imprimante Laser au format A3. Dans cet exemple, le facteur de réduction vaut exactement 2.

Si ce facteur nuit à l'exploitation du document, il faut envisager un autre type de traceur.

Le traceur ou plotter est capable de traiter les plus grands formats jusqu'à A0.

Le fonctionnement du plotter est exactement identique à celui du dessinateur et de son crayon.

Les plotters sont en général assez lents, en comparaison avec les impressions sur imprimantes Laser. Les coûts d'acquisition de l'exploitation et de maintenance sont aussi beaucoup plus élevés que ceux auxquels on peut s'attendre avec des imprimantes Laser.

L'acquisition d'un traceur grand format est donc réservée à des cas exceptionnels où l'on doit expressément travailler sur des grands documents.

Les imprimantes sont liées au micro-ordinateur par des interfaces parallèles : les données sont transférées en parallèle, octet par octet. Les traceurs sont liés au micro-ordinateur par des interfaces série : les données sont transférées en série.

L'interface série est conforme à la norme RS 232.

Le signal est constitué d'un bit de départ (bit start), de 7 ou 8 bits de données, d'un bit de parité optionnel et d'un ou deux bits de stop – bits de stop –.

Les vitesses de transmission standard sont de 110, 300, 1 200, 2 400, 4 800, 9 600, et 19 200 bits par seconde.

## 6 Interface d'entrée/sortie

Le micro-ordinateur se transforme assez facilement en un automate programmable à partir de l'instant où il dispose de cartes d'entrée/sorties.

Ces cartes peuvent être de deux types différents : les cartes d'entrées/sorties numériques et les cartes d'entrées/sorties analogiques.

### *6.1 Cartes d'entrées/sorties numériques*

Ces interfaces sont chargées de mémoriser l'état d'un certain nombre d'entrées.

Ces états peuvent être transférés à l'unité centrale pour une exploitation quelconque.

L'opération inverse est également possible : l'écriture d'un certain nombre d'états de la CPU vers les sorties.

Le nombre d'entrées/sorties n'est limité que par la conception de la carte d'interface. La conception d'une carte 32 entrées – 32 sorties ne pose pas de problème majeur.

D'un point de vue électrique, les entrées/sorties peuvent être :
- compatibles TTL,
- boucles de courant 0-20mA ou 4-20mA,
- opto-isolées,
- à relais,
- etc.

Ces cartes relient le « monde extérieur » au micro-ordinateur. La conception d'un automate se résume à l'écriture d'un programme d'acquisition des valeurs d'entrée, d'un certain nombre de calculs et d'écriture des valeurs de sortie.

### *6.2 Cartes d'entrée/sortie analogiques*

La plupart des capteurs délivrent des signaux analogiques : capteurs de température, force, pression, position, etc.

Les grandeurs analogiques sont préalablement converties en numériques pour transiter sur le bus du PC et être traitées par la CPU.

De la même manière, il peut être intéressant de disposer en sortie d'une information analogique destinée par exemple à un moteur ; pour les entrées, on dispose donc de convertisseurs analogiques-numériques ; pour les sorties, les convertisseurs sont du type numériques-analogiques.

Les caractéristiques essentielles de ces cartes sont la résolution : largeur du mot sur lequel le résultat est exprimé, en général compris entre 8 et 16 bits ; cadence d'échantillonnage, variable en fonction du type de convertisseur mis en service. On se référera au chapitre traitant des convertisseurs.

## 7 Interface réseau

Les cartes d'entrée/sortie sont un premier exemple de connexion du micro-ordinateur au monde extérieur. Un deuxième type de connexion est intéressant : la connexion des micro-ordinateurs entre eux.

On peut distinguer deux types de connexion, les connexions locales ou à distance.

### *7.1 Réseaux locaux LAN : Local Area Network*

Un réseau local permet de connecter un groupe de micro-ordinateurs et éventuellement un certain nombre de périphériques comme les imprimantes ou les traceurs.

Le meilleur exemple de réseau local est le réseau Ethernet qui, sur un câble coaxial assez classique, autorise des transmissions à 10 M bits/s.

Le principal intérêt du réseau local réside dans le partage des ressources ; chaque station de travail peut être équipée des ressources minimales – mémoire de masse par exemple – et une des stations peut être dédiée au stockage.

Les traceurs, extrêmement coûteux, peuvent être partagés par plusieurs utilisateurs.

Les utilisateurs connectés au réseau local peuvent échanger des fichiers ; si ces fichiers sont de simples messages entre les utilisateurs, on parle alors de courrier électronique.

### *7.2 Communication à distance*

Pour des communications à plus longue distance, le micro-ordinateur est équipé d'un modem « modulateur - démodulateur ».

Une ligne téléphonique suffit alors pour interconnecter plusieurs machines.

Les vitesses de transfert sont beaucoup plus faibles que celles des réseaux locaux, de l'ordre de quelques dizaines de K bits/seconde.

L'intérêt de ce type de connexion est évident ; il peut permettre à plusieurs entreprises de coopérer sur un même projet. Il permet aussi à tout utilisateur d'accéder à de très importantes bases de données, via Internet par exemple.

Chapitre 5

# CONSOLE DE SÉLECTION VIDÉO À MICROCONTRÔLEUR

**SOMMAIRE**


- Sélection de la fonction à accomplir
- Choix et description des composants
- Schéma de principe
- Manuel d'utilisation


*Même si l'on ne dispose pas d'une installation quasi professionnelle, il n'y a rien d'extraordinaire à rencontrer, à proximité du petit écran, un ou deux magnétoscopes, et/ou un caméscope, un décodeur Canal Plus et un récepteur de télévision par satellite. Comment relier tous ces périphériques au téléviseur ? Voilà qui n'est pas simple. En général, les téléviseurs ne sont équipés que d'une et une seule prise dite Péritel, permettant le branchement d'un et d'un seul périphérique.*

## Sélection de la fonction à accomplir

Quotidiennement nous utilisons donc un ou plusieurs de ces périphériques et la configuration de l'ensemble peut changer en fonction du type d'application. Il faut donc pouvoir reconfigurer l'installation rapidement et sûrement, en effectuant un minimum de traitement. On doit pouvoir :
- choisir la source à visualiser,
- intercaler un décodeur,
- enregistrer un signal quelconque (le signal visualisé ou un autre).

Dans l'installation décrite précédemment (magnétoscope, décodeur, récepteur TV–Sat, etc.), il manque donc un maillon. Ce maillon, c'est tout simplement un aiguillage, capable d'envoyer toutes les voies d'entrée sur toutes les voies de sortie, sans limitations. Cet aiguillage est appelé commutateur Péritel ou console de commutation Péritel.

La console de commutation, décrite dans les pages qui suivent, n'est rien d'autre qu'un aiguillage assurant les interconnexions entre les divers périphériques.

Avant tout il convient de définir le commutateur et sa fonctionnalité. La première question concerne le nombre de périphériques à connecter.

## Choix et description des composants

La première réflexion concerne les circuits de commutation analogique ayant une largeur de bande suffisante pour un signal vidéo : au moins 5 MHz.

Les circuits intégrés Philips TDA 8440 permettent la sélection d'une voie parmi deux. Si l'on veut commuter quatre voies, deux circuits sont nécessaires ; six voies, trois circuits, etc. Ceci conduit à l'élaboration d'un appareil ne comportant pas moins de 10 TDA 8440, capable de gérer 4 périphériques. La multiplicité des composants a le double inconvénient du coût et de l'encombrement.

Pour cet appareil, un circuit plus complexe, gérant quatre entrées et trois sorties, a été retenu : c'est le CXA1114 Sony.

On devine donc qu'en utilisant un seul de ces circuits, on est capable de gérer trois périphériques, ce que montre le schéma synoptique.

### 1 Schéma synoptique du commutateur

Le schéma synoptique du commutateur est représenté à la figure 5.1 ; les entrées/sorties Péritel 1 à Péritel 3, correspondant aux trois périphériques, et Péritel 4 à la liaison avec le téléviseur. La structure du commutateur électronique utilisé ne permet pas de banaliser les trois entrées/sorties. En conséquence, Péritel 1 est une entrée seulement et Péritel 2 et 3 sont des entrées/sorties. Cet état de chose limite un peu les performances de l'appareil mais ceci est sans grande conséquence. En effet, pour certains périphériques, la notion de renvoi d'une source n'a pas de sens. Un récepteur satellite, par exemple, fournit des signaux vidéo et audio mais n'en reçoit pas. Péritel 1 doit donc être considérée comme une entrée seulement. Les trois commutateurs 0, 1 et 2 permettent de renvoyer, vers la prise Péritel concernée, l'une des quatre voies d'entrée. Ceci signifie que l'on peut bien sûr sélectionner la voie vidéo visualisée mais aussi sélectionner la voie vidéo envoyée vers Péritel 2 ou Péritel 3. Dans un cas concret, cette configuration permet par exemple de relier un magnétoscope sur Péritel 2, un décrypteur sur Péritel 3 et soit un récepteur satellite, soit un deuxième magnétoscope en lecture seulement sur Péritel 1. Ceci permet donc d'enregistrer en clair un programme crypté et de visualiser soit le dit programme soit un autre programme, qu'il soit terrestre ou satellite. Toutes les configurations sont possibles, la seule restriction étant Péritel 1 en entrée seulement.

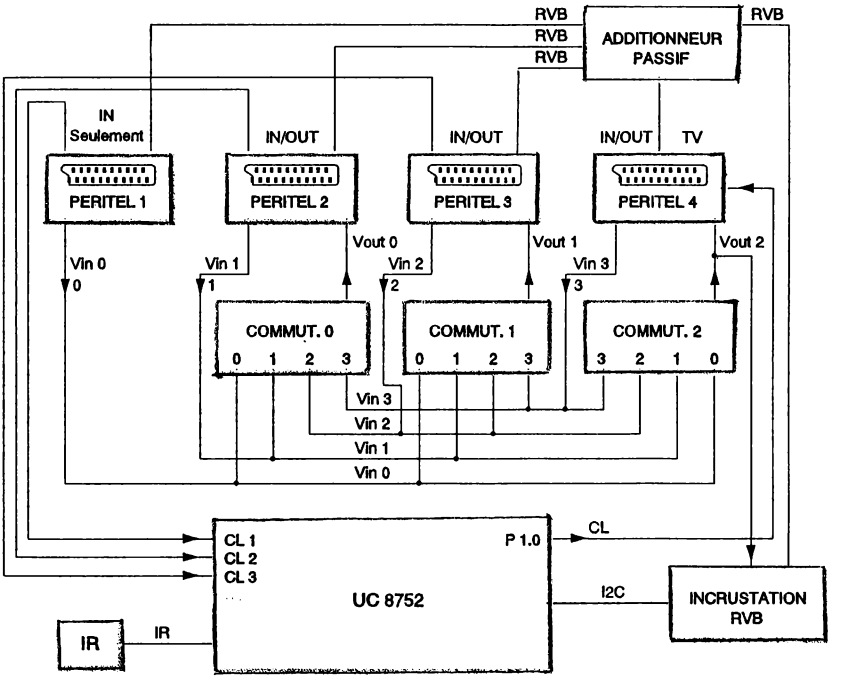



Fig. 5.1
Schéma synoptique du commutateur vidéo.

Sur le synoptique de la figure 5.1, on remarque que toutes les entrées RVB commutation rapide des Péritel 1 à 3 sont additionnées et envoyées vers le téléviseur. Cet additionneur permet la restitution des divers menus en provenance des périphériques : satellite, décrypteurs ou magnétoscope. A ces informations issues des périphériques viennent s'ajouter des informations RVB locales. En effet, pour minimiser le coût de cet appareil, on doit éviter au maximum tous les systèmes d'affichage en face avant, soit à LED ou LCD sept segments ou pas. On a donc recours à une incrustation sur l'écran du téléviseur qui ne coûte que quelques lignes de programme. Cette solution a en outre l'avantage de présenter les informations de manière claire et précise, ce qui est impensable sur des afficheurs sept segments. Dans le même ordre d'idée que précédemment, on cherche à minimiser le nombre de commandes en face avant. Une télécommande infrarouge supprime toutes les commandes mécaniques (clavier) et règle le problème de manière élégante. Ceci donne un aperçu des possibilités de cet appareil et de sa conception. La description sommaire des deux composants principaux est essentielle avant d'aborder le schéma de principe.

## 2 Le circuit commutateur SONY CXA 1114

Le schéma synoptique interne du circuit de commutation est représenté à la figure 5.2.

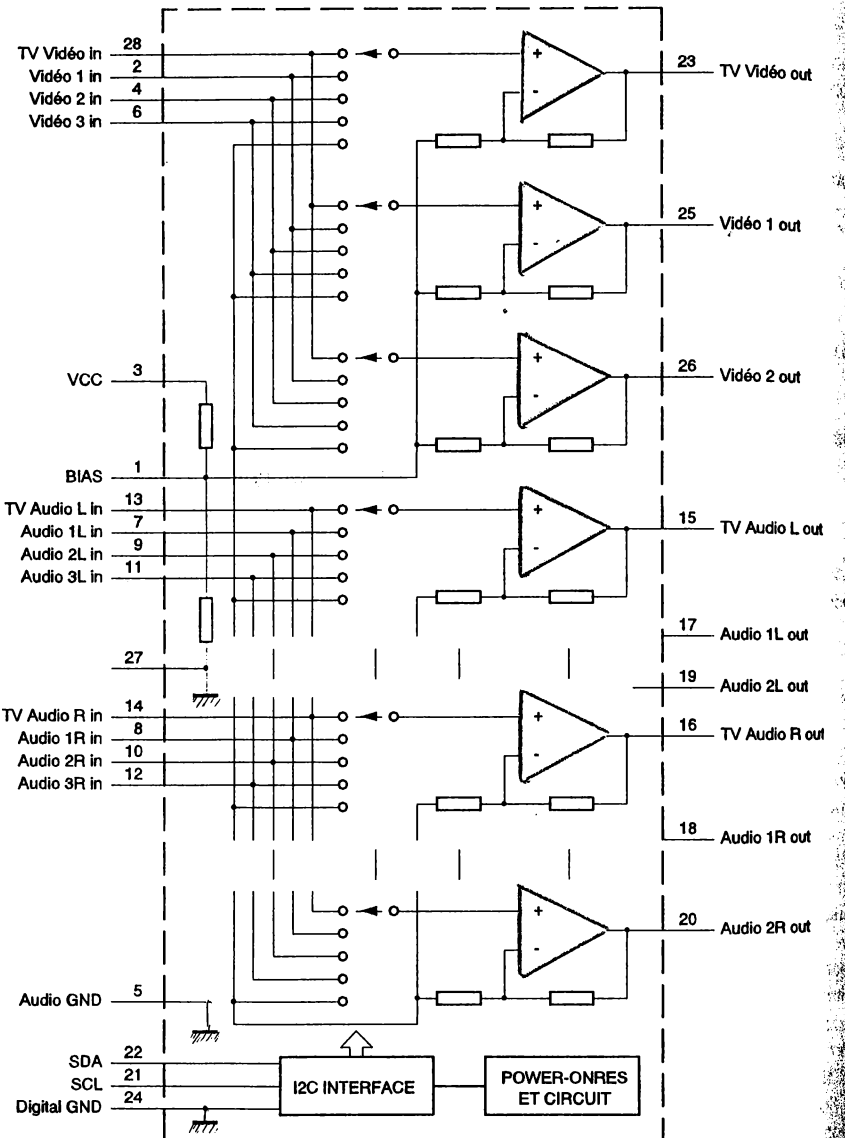



Fig. 5.2
Schéma synoptique interne du circuit de commutation vidéo + audio CXA 1114 Sony.

Une voie se compose d'une voie vidéo et de deux voies audio, soit une vidéo accompagnée d'une audio stéréo. Le circuit Sony CXA 1114 est donc un aiguilleur capable de gérer quatre entrées et trois sorties. Les caractéristiques de ce circuit sont excellentes. La bande passante vidéo s'élève à 15 MHz tandis que la bande passante audio s'élève à 100 KHz. Sur la voie vidéo, le gain vaut 6 dB, ce qui signifie qu'aucun amplificateur externe ne sera nécessaire pour réaliser l'adaptation à 75 ohms. Ce circuit ne requiert qu'une alimentation + 9 V et consomme 50 mA. Les commutateurs internes sont actionnés via un bus I2C, ce qui limite à 24 le nombre de broches du boîtier. Bien que la fonction accomplie soit complexe et le nombre d'entrées/sorties important, le boîtier reste de taille compacte. Pour ce circuit, la sous-adresse I2C vaut 90 Hex. Il existe une deuxième version de ce circuit, le CXA 1434, totalement compatible avec le CXA 1114 pour lequel la sous-adresse vaut 96 Hex.

| S | adresse | A | DATA0 | A | DATA1 | A | DATA2 | A | P |
|---|---|---|---|---|---|---|---|---|---|

S : Start condition, condition de départ.
A : Acknowledge émis par l'esclave.
P : Stop condition, condition d'arrêt.

Structure des registres de contrôle DATA0 à DATA2 :

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| * | VM | VS1 | VS0 | * | AM | AS1 | AS0 |

* Les bits b7 et b3 ne sont pas définis.

À la mise sous tension, tous les bits sont à zéro.

| VM | VS1 | VS0 | SORTIE |
|---|---|---|---|
| 0 | * | * | aucune |
| 1 | 0 | 0 | TV VIDEO IN |
| 1 | 0 | 1 | VIDEO 1 IN |
| 1 | 1 | 0 | VIDEO 2 IN |
| 1 | 1 | 1 | VIDEO 3 IN |

| AM | AS1 | AS0 | SORTIE |
|---|---|---|---|
| 0 | * | * | aucune |
| 1 | 0 | 0 | TV AUDIO IN |
| 1 | 0 | 1 | AUDIO 1 IN |
| 1 | 1 | 0 | AUDIO 2 IN |
| 1 | 1 | 1 | AUDIO 3 IN |

Le tableau de la figure 5.3 résume parfaitement le message I2C envoyé au circuit et la manière dont celui-ci l'interprète. Les trois commutateurs internes sont paramétrés grâce aux données DATA0 à DATA2 constituées chacune d'un octet. La définition de cet octet est jointe à la figure 5.2. Logiquement, la voie audio suit la voie vidéo, ce qui simplifie énormément le tableau de commutation qui se résume à cinq cas :

- Mute data = 0,
- Voie 0 data = 44 Hex,
- Voie 1 data = 55 Hex,
- Voie 2 data = 66 Hex,
- Voie 3 data = 77 Hex.

| S | Slave address 90H/96H | A | DATA0 | A | DATA1 | A | DATA2 | A | P |
|---|---|---|---|---|---|---|---|---|---|

S : Start condition
A : Acknowledge emitted by slave (CXA1114/CXA1434)
P : Stop condition

Structure of Respective Control Registers (DATA 0 to 2)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| x | VM | VS1 | VS0 | x | AM | AS1 | AS0 |

* b7, b3, undefined
* At Power On all bits turn to "0". (Power On Reset function)

Video switch control

| VM | VS1 | VS0 | Output pin |
|---|---|---|---|
| 0 | x | x | Mute (blanking) |
| 1 | 0 | 0 | TV Video in |
| 1 | 0 | 1 | Video 1 in |
| 1 | 1 | 0 | Video 2 in |
| 1 | 1 | 1 | Video 3 in |

Audio switch control

| AM | AS1 | AS0 | Output pin |
|---|---|---|---|
| 0 | x | x | Mute |
| 1 | 0 | 0 | TV Audio in |
| 1 | 0 | 1 | Audio 1 in |
| 1 | 1 | 0 | Audio 2 in |
| 1 | 1 | 1 | Audio 3 in |

Fig. 5.3
Structure du message I2C interprété par le CXA 1114 Sony.

## 3 Le circuit SAA 5244 A

Dans les lignes qui précèdent, il est précisé que le menu de configuration sera incrusté sur l'écran.

On doit donc sélectionner un circuit d'incrustation et d'affichage RVB. Il existe de nombreux circuits capables d'accomplir cette mission, ils sont en général très différents, donc incompatibles entre eux, et l'intégration des fonctions est plus ou moins poussée. Ce qui veut dire, en clair, que certains circuits d'apparence simple requièrent un grand nombre de composants périphériques. A contrario, certains circuits sont d'apparence plus complexe car l'intégration est plus poussée. Toutes les fonctions nécessaires à la mise en œuvre du circuit sont regroupées dans le même boîtier. Tel est le cas de la circuiterie de synchronisation assez rarement

présente. Tel n'est pas le cas du SAA.5244 A que nous avons sélectionné.

Le circuit SAA 5244 A est un circuit décodeur–afficheur télétexte compatible CEEFAX. Ce qui revient à dire que les deux fonctions fondamentales de ce circuit sont l'acquisition des informations et l'affichage. Il est possible de se servir de l'une ou l'autre de ces fonctions ou les deux. Dans ce cas, les deux fonctions affichage et décodage télétexte sont exploitées. Les informations relatives à la page à afficher sont transmises du microcontrôleur au circuit via le bus I2C. La mise en œuvre du circuit est extrêmement simplifiée, il suffit d'un oscillateur à quartz à 27 000 MHz et d'une interface de sortie à transistors sur les signaux de RVB et commutation rapide. Le circuit a finalement deux possibilités intéressantes :
– soit un verrouillage sur un signal vidéo incident,
– soit un fonctionnement autonome en générant une synchronisation composite.

En mode autonome, le signal de synchronisation est disponible sur la broche STTV – Synchro To TV –.

Dans cette application, seul le mode verrouillage sur un signal vidéo incident est utilisé.

Cette vidéo sera soit la vidéo terrestre issue du téléviseur, soit la vidéo en provenance d'un périphérique et bien sûr tout cela s'effectue de manière automatique. Dans la famille SAA 524*, il existe d'autres circuits.

Le SAA5 244 A a été retenu pour sa caractéristique d'autonomie.

Tous les autres circuits requièrent une mémoire externe. Qui dit mémoire supplémentaire dit augmentation du coût des composants et du circuit imprimé. Or ce commutateur a été conçu avec l'impératif principal du coût minimum.

## 4 Choix du microcontrôleur et de la structure

Pour gérer le bus I2C, la meilleure solution consiste dans l'emploi d'un microcontrôleur. Dans cette famille, le choix est vaste.

Il n'est pas dans l'intention des auteurs de passer en revue les avantages et inconvénients de tel ou tel microcontrôleur, et pour le commutateur Péritel, le 8051, qui a été décrit dans le chapitre précédent, a été retenu. Bien sûr, il eut été possible d'employer le 8052 AH Basic qui a l'avantage d'une grande souplesse d'emploi et requiert un minimum de matériel pour la phase de développement du programme. Mais le 8052 AH Basic a un inconvénient majeur : sa lenteur due au Basic interprété.

Il existe trois configurations dont les avantages et inconvénients ont été signalés au chapitre 4 :

– microcontrôleur à EPROM intégré, effaçable ou OTP de type 8751 ou 8751 OTP,

– microcontrôleur sans EPROM et mémoire programme externe 8031 + 2764,
– microcontrôleur sans EPROM avec mémoire programme et mémoire RAM externes.

Dans le premier cas, cette solution est intéressante au moins pour deux raisons : encombrement et coût dans le cas de l'OTP (one time programming), c'est la solution industrielle pour les petites et moyennes séries.

Le deuxième cas est particulièrement intéressant pour la phase du développement configuration, utilisable aussi en petite et moyenne série lorsqu'il n'existe pas d'impératif d'encombrement.

La dernière configuration est d'un point de vue hardware la plus complexe, c'est aussi la plus coûteuse en circuit imprimé.

Mais cette configuration présente au moins l'avantage de pouvoir travailler avec un espace RAM non limité à 256 ou 128 octets.

Pour ce commutateur, il a été ajouté un impératif supplémentaire : un circuit imprimé simple face. Dans ces conditions, il faut choisir entre les deux premières solutions. C'est finalement la première solution, le 8751 OTP, qui a été retenue pour simplifier au maximum le tracé du circuit et limiter sa taille.

## Schéma de principe

Le schéma de principe est représenté à la figure 5.4 (pages 182 et 183). La lecture de ce schéma est assez simple pourvu que l'on dissocie les trois sous-ensembles primordiaux : microcontrôleur, incrustateur et commutateur.

### 1 Microcontrôleur

Le microcontrôleur reçoit les informations infrarouges en provenance du récepteur TEMIC TFMS 5360. Le microcontrôleur récupère les informations à la broche 12 INT0 et elles sont interprétées par le programme.

Le programme est prévu pour des informations au format RC5.

Le programme utilise une référence de temps établie à partir du quartz à 11.0592 MHz. Ceci signifie que cette valeur ne doit pas être modifiée. Les ports P1.6 et P1.7 sont classiquement affectés au bus I2C. Sur le port P2, on récupère les informations commutation lente en provenance des trois périphériques. Les trois signaux CL1 à CL3 sont envoyés sur P2.0 à P2.2. À l'origine, cette interface avait été prévue de manière à ce qu'un périphérique puisse automatiquement prendre la main dès sa mise sous tension. Par exemple, lorsqu'un magnétoscope passe en lecture, il active sa commutation lente, celle-ci est détectée par le micro-

contrôleur qui actionne le commutateur ad-hoc du CXA 1114 et les signaux sont automatiquement envoyés vers le téléviseur. Cette procédure automatique n'a pas que des avantages, notamment dans le cas d'un récepteur satellite, ou dans la fonction copie magnétoscope ou caméscope vers magnétoscope.

En effet, on peut vouloir enregistrer une émission satellite ou effectuer une copie sans pour autant visualiser l'opération en cours. Il faut alors que cette fonction soit débrayable. Pour simplifier au maximum le commutateur et son usage, ces signaux ne sont pas exploités. Les informations CL1 à CL3 sont donc ignorées par le programme. En conséquence, les éléments suivants peuvent être omis : D25, D26, D27, D10, D15 et D20 ainsi que R67 à R72.

## 2 Incrustation

Autour du SAA 5244 A, on retrouve premièrement l'oscillateur à 27 000 MHz. Le circuit se verrouille sur le signal vidéo reçu sur la broche 8 et les informations RVB et BLAN sont en phase avec le signal incident. Les transistors Q1 à Q5 font office de tampon. L'additionneur RVB entre les trois périphériques et l'incrustateur local est confié aux diodes D1 à D24.

## 3 Commutateur CXA 1114

L'interface avec les entrées/sorties Péritel se résume simplement à la présence de condensateurs de blocage DC et à la présence de trois transistors Q6 à Q8. Les sorties vidéo sont adaptées 75 ohms et l'amplitude crête à crête fixée à 1 V. Ceci achève la description du schéma de principe qui, grâce à l'emploi de composants hautement intégrés, est finalement assez simple.

## 4 Télécommande IR

La télécommande utilisée est référencée : VISA, IR 180.

Les signaux émis sont conformes au code RC5 qui est un code biphase (voir chapitre 9).

Le signal est injecté à l'entrée interruption du microcontrôleur et la base de temps interne est employée pour décoder le message émis.

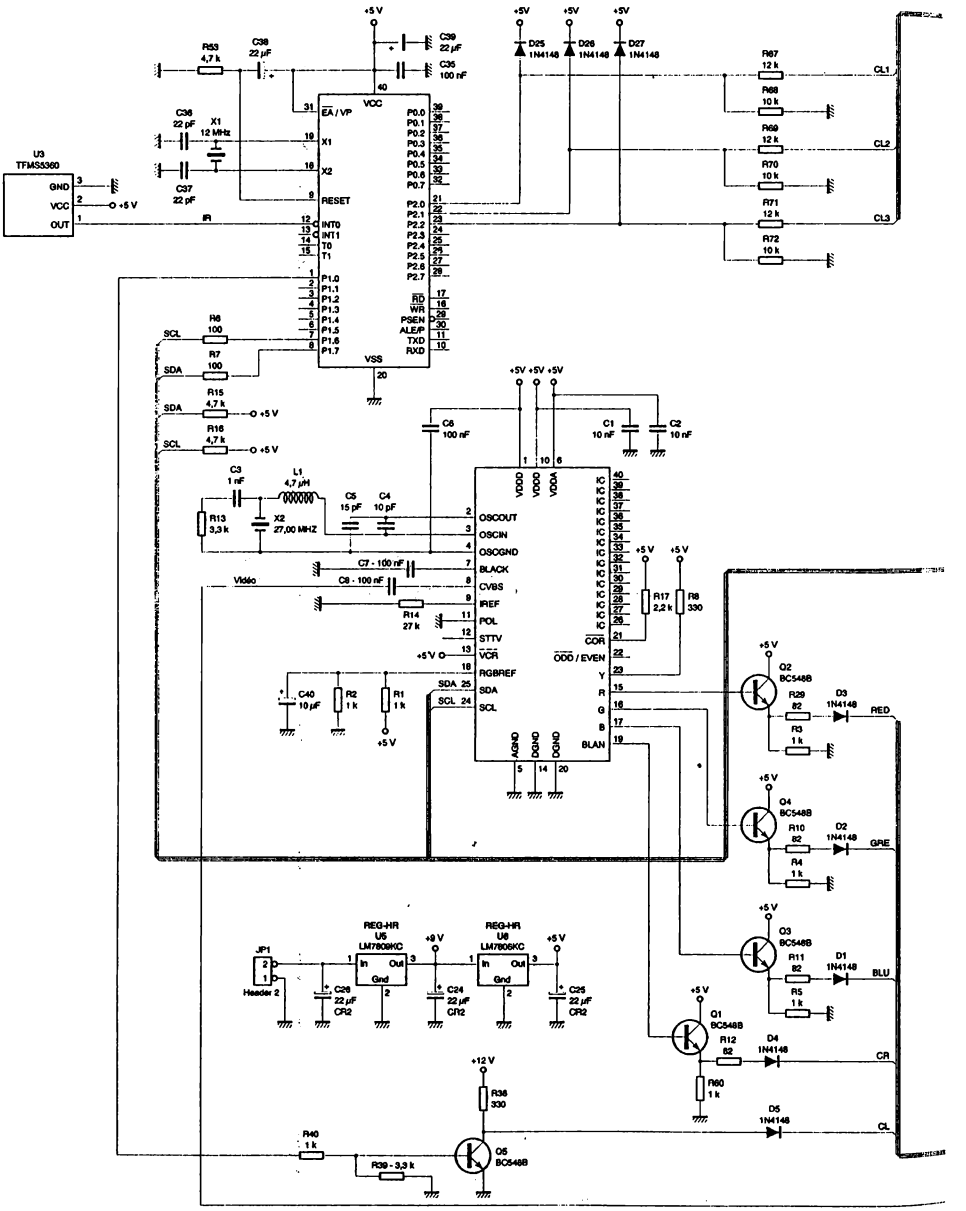



Fig. 5.4
Schéma de principe de la console de sélection vidéo à microcontrôleur.

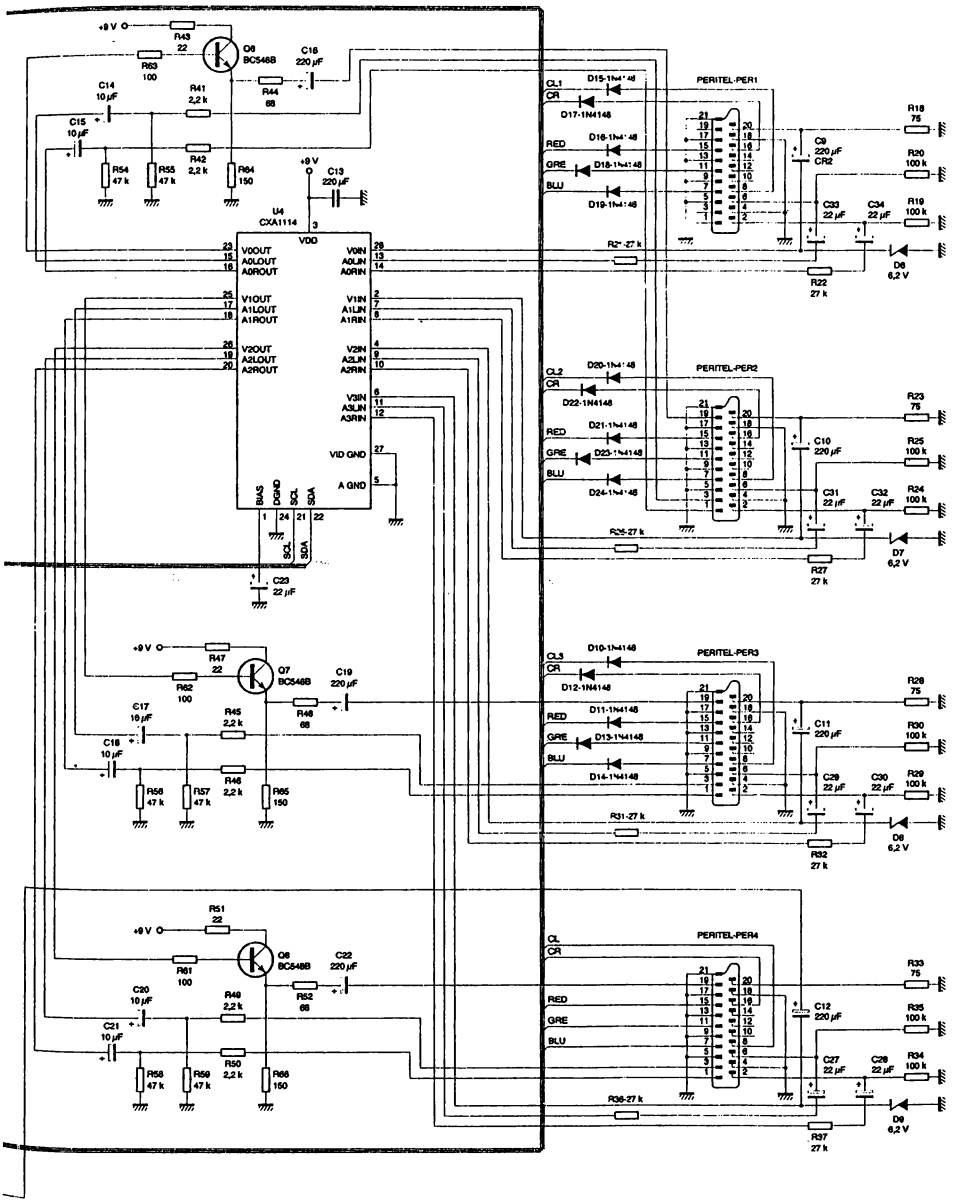

+9 V
R43 22
R63 100
Q6 BC548B
C16 220 µF
C14 10 µF
R41 2,2 k
R44 68
C15 10 µF
R54 47 k
R55 47 k
R64 150
R42 2,2 k
+9 V
C13 220 µF
U4 CXA1114
VDD
V0OUT A0LOUT A0ROUT
V0IN A0LIN A0RIN
V1OUT A1LOUT A1ROUT
V1IN A1LIN A1RIN
V2OUT A2LOUT A2ROUT
V2IN A2LIN A2RIN
V3IN A3LIN A3RIN
VID GND
A GND
BIAS DGND SCL SDA
C23 22 µF
CL1 CR RED GRE BLU
D15-1N4148
D17-1N4148
D16-1N4148
D18-1N4148
D19-1N4148
PERITEL-PER1
C9 220 µF CR2
C33 22 µF
C34 22 µF
R18 75
R20 100 k
R19 100 k
R2-27 k
R22 27 k
D6 6,2 V
CL2 CR RED GRE BLU
D20-1N4148
D22-1N4148
D21-1N4148
D23-1N4148
D24-1N4148
PERITEL-PER2
C10 220 µF
C31 22 µF
C32 22 µF
R23 75
R25 100 k
R24 100 k
R26-27 k
R27 27 k
D7 6,2 V
+9 V
R47 22
R62 100
Q7 BC548B
C19 220 µF
C17 10 µF
C18 10 µF
R45 2,2 k
R48 68
R46 2,2 k
R58 47 k
R57 47 k
R65 150
CL3 CR RED GRE BLU
D10-1N4148
D12-1N4148
D11-1N4148
D13-1N4148
D14-1N4148
PERITEL-PER3
C11 220 µF
C29 22 µF
C30 22 µF
R28 75
R30 100 k
R29 100 k
R31-27 k
R32 27 k
D8 6,2 V
+9 V
R51 22
R61 100
Q8 BC548B
C22 220 µF
C20 10 µF
C21 10 µF
R49 2,2 k
R52 68
R50 2,2 k
R56 47 k
R59 47 k
R66 150
CL CR RED GRE BLU
PERITEL-PER4
C12 220 µF
C27 22 µF
C28 22 µF
R33 75
R35 100 k
R34 100 k
R36-27 k
R37 27 k
D9 6,2 V

Fig. 5.4 (suite)

## Manuel d'utilisation

Un examen approfondi du programme serait trop long pour figurer dans ce chapitre.

Les lignes qui suivent donnent une idée des critères établis pour l'écriture du programme.

La télécommande génère des trames infrarouges au format RC5 étendu, avec soit le code système d'un téléviseur (code 0), soit le code système d'un magnétoscope (code 5). Le logiciel traite indifféremment les trames comportant les codes système 0 ou 5.

Le logiciel de traitement du commutateur Péritel, interprète 7 touches de la télécommande, sur les 39 disponibles, pour la fonction gestion des entrées/sorties :

– Touche « TV mode » : un appui sur cette touche permet de faire disparaître les textes incrustés par le commutateur sur l'écran du téléviseur.

– Touche « MIX mode » : un appui sur cette touche permet de faire apparaître le menu de configuration en surimpression de l'image.

– Touche « TXT mode » : un appui sur cette touche permet de faire apparaître le menu de configuration du commutateur Péritel. Le centre de l'image du téléviseur est dans ce cas remplacé par un carré noir dans lequel les textes du menu de configuration sont incrustés.

– Le commutateur est composé de 4 prises Péritel, l'une d'entre elles est dédiée au téléviseur (Péritel 4). Les Péritel 2 et 3 sont utilisables en entrée et en sortie, la Péritel 1 est utilisable uniquement en sortie. Les prises programmables sont les prises 2, 3 et 4.

– Les touches « UP » et « DOWN » permettent de sélectionner la prise Péritel à programmer. Une flèche en début de ligne identifie la prise Péritel que l'on programme. Les touches « + » et « – » permettent de sélectionner la source vidéo que l'on dirige vers la prise Péritel à programmer.

Le commutateur Péritel utilise, pour les incrustations, le SAA 5244 A, un circuit qui comprend un décodeur télétexte. Sans modifications hardware de la carte, il est simple d'intégrer cette seconde fonction : un décodeur télétexte.

Le programme inclus dans le microcontrôleur intègre cette fonction. Les commandes de recherche de pages particulières télétexte sont accessibles directement à partir de la télécommande IR.

Cette application est un bon exemple d'utilisation et de mise en œuvre d'un microcontrôleur.

Le microcontrôleur reçoit une information codée au format RC5, la décode, l'interprète et génère une action se traduisant par l'émission d'un message au format I2C.

On remarquera que le nombre d'entrées/sorties disponibles sur les ports du microcontrôleur est suffisant pour gérer un clavier supplémentaire si le besoin s'en faisait sentir.

Dans toute application mettant en œuvre un microcontrôleur, on cherchera à simplifier au maximum la circuiterie. En conséquence, le microcontrôleur devra traiter un maximum d'opérations ou de fonctions.

# 2 LES CIRCUITS ÉLECTRONIQUES COMPLEXES


6. Sources stabilisées et alimentation des systèmes
7. Conversion analogique-numérique et numérique-analogique
8. Synthétiseur de fréquence
9. Modems et transmissions numériques
10. Circuits d'interface pour bus et réseaux
11. Filtres à capacités commutées et filtres numériques
12. Application : carte d'acquisition vidéo pour PC


Étant donné l'utilisation croissante en électronique des circuits présentant un haut niveau d'intégration, c'est-à-dire comportant sur le même morceau de silicium (*chip*) un nombre important de fonctions élémentaires, il est indispensable que l'utilisateur ou le concepteur connaisse les grandes familles de circuits intégrés disponibles chez les constructeurs et assimile les concepts de base permettant de comprendre les conditions et les limites d'utilisation de ces circuits. Dans la pratique, le concepteur se réfère aux manuels de description édités par les fabriquants de circuits. Ces ouvrages (*Data handbooks*) sont en général remarquablement écrits, toujours en langue anglaise cependant. Ils peuvent dérouter le lecteur parce que les principes généraux sont rarement rappelés et parce qu'il n'est pas toujours facile de faire le tri entre l'information indispensable et l'information moins utile. De plus, ces manuels sont proposés par les constructeurs et ne décrivent que les produits de ces dits constructeurs.
On peut citer sans prétendre à l'exhaustivité : Philips, Motorola, Texas, Intel, Sony, Thomson, Analog Devices, Burr Brown, National, Siemens, Maxim, Fujitsu, etc.
Toute classification est imparfaite et celle que nous proposons est à la fois incomplète et subjective. Le lecteur nous pardonnera si un certain nombre de fonctions complexes ne sont pas traitées dans ce chapitre, comme par exemple les multiplicateurs câblés, les commandes de moteurs, et bien d'autres. Le champ de l'électronique est maintenant si vaste qu'une encyclopédie serait à peine

suffisante pour décrire tous les circuits. Nous avons donc choisi de classer les circuits en fonction de six fonctions de caractère général demandées dans la conception des systèmes électroniques :

- alimenter les composants,
- passer du monde analogique au monde numérique,
- synthétiser des signaux de fréquence donnée,
- moduler des signaux alternatifs,
- relier des circuits à des bus ou à des réseaux,
- filtrer en profitant des avantages des techniques numériques.

Les microprocesseurs, les microcontrôleurs, les processeurs de signaux et les circuits programmables ont été décrits dans d'autres chapitres de l'ouvrage.

Chapitre 6

# SOURCES STABILISÉES ET ALIMENTATION DES SYSTÈMES

## SOMMAIRE


- Utilisation des sources de tension stabilisées
- Principe de fonctionnement des sources de tension stabilisées
- Exemple simple d'alimentation
- Alimentation à découpage et convertisseur continu–continu
- Caractéristiques des régulateurs et critères de choix


*L'objectif de ce chapitre est double : d'une part montrer l'intérêt des régulateurs de tension pour alimenter les composants d'un système, d'autre part définir les caractéristiques principales des produits du commerce et donner au lecteur des critères de choix. Deux cas se présentent dans la pratique. Dans le premier cas, le concepteur dispose d'une source de tension continue ou de plusieurs sources continues pour alimenter son système. Il suffit alors de répartir la puissance disponible en respectant un certain nombre de règles élémentaires qui seront exposées dans ce chapitre.*
*Dans le deuxième cas, le concepteur ne dispose que du secteur électrique du réseau de distribution (220 V alternatif, 50 Hz en France). Dans ce cas, il est nécessaire de créer les tensions continues demandées pour l'alimentation des circuits intégrés. Les alimentations sont de type linéaire ou de type à découpage ; il est donc nécessaire de donner les avantages et les inconvénients des deux familles.*
*Une importante évolution du monde électronique est le développement considérable des systèmes autonomes fonctionnant à l'aide de piles ou de batteries rechargeables.*

## Utilisation des sources de tension stabilisées

Dans un premier temps, plaçons-nous dans le cas suivant : il est nécessaire d'alimenter un certain nombre de cartes électroniques ou de modules à partir d'une source de tension continue unique. Il faut généralement disposer de plusieurs valeurs positives ou négatives. Par exemple, les circuits logiques en technologie CMOS

demandent une tension de + 5 V. Les circuits analogiques utilisent des tensions positives et négatives de 6 V, parfois de 12 V. Comme les sources de tension disponibles ne sont pas toujours celles qu'il est nécessaire d'utiliser pour alimenter les circuits, il faut donc les créer à partir des valeurs dont on dispose. De plus, il est indispensable de répartir convenablement les tensions vers les différentes cartes du système. Dans tous les cas, la source de tension alimentant un circuit intégré doit présenter les caractéristiques suivantes :

- la valeur doit être stable dans le temps,
- elle doit être indépendante du courant consommé,
- elle doit être peu dépendante des variations de température ambiante,
- les ondulations résiduelles (résidus du secteur 50 Hz) doivent être minimisées.

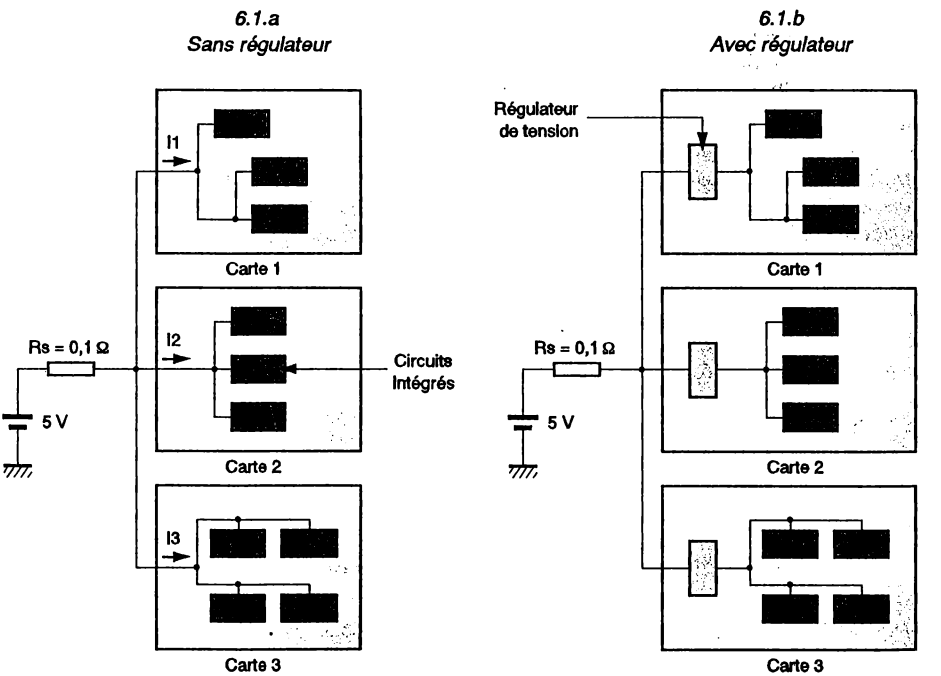



Fig. 6.1
Alimentation d'un système électronique.

Dans les cas contraires, les circuits fonctionnent avec des performances réduites ou même ne réalisent plus leur fonction.

Deux solutions s'offrent alors : la première est de distribuer la tension d'alimentation à toutes les cartes et composants du système, la seconde est de créer sur chaque carte les tensions nécessaires. En fait, la première solution est très mauvaise et ne doit jamais être utilisée. L'exemple donné figure 6.1 permet de comprendre l'intérêt de créer les tensions d'alimentation sur chaque carte et introduit les circuits réalisant cette fonction. Ces circuits sont appelés régulateurs de tension. Dans la configuration 6.1.a, la tension 5 V est simplement distribuée aux différents circuits. Il est alors facile de constater qu'une variation de courant consommé par une carte a pour effet une variation de la tension appliquée. Pour cela, il suffit de faire intervenir la résistance *Rs*. Cette résistance est l'impédance de sortie de la source de tension. Elle inclut également les diverses résistances de contact qui sont inévitablement présentes dans l'interconnexion. Une valeur de 0,1 Ω est vraisemblable. Si le courant $I_1$ consommé par la carte 1 varie de 1 A, alors la tension appliquée diminue de 100 mV. Il suffit d'appliquer la loi d'Ohm pour établir ce résultat (1 A × 0,1 Ω). Ce phénomène introduit donc un couplage entre la carte 1 et les autres cartes du système puisque tous les composants voient leur tension d'alimentation varier de 100 mV. La **réjection** des circuits vis-à-vis des variations de la tension d'alimentation est peut être suffisamment bonne pour que les conséquences soient négligeables mais ce n'est pas toujours le cas. De plus, des valeurs plus élevées de l'impédance série aggravent le phénomène.

Pour éviter ces inconvénients, il est conseillé de choisir une architecture de type 6.1.b. Des circuits spéciaux sont placés en entrée de chaque carte. Ils sont appelés régulateurs de tension. Ces circuits permettent en plus de créer les valeurs exactes des tensions nécessaires.

## Principe de fonctionnement des sources de tension stabilisées

Détailler la boîte noire formée par le régulateur de tension permettra au lecteur de se familiariser avec les grandeurs caractéristiques de ces circuits et de guider son choix parmi la grande quantité de produits offerts par les fabricants. De plus, le régulateur de tension est un excellent exemple de circuit électronique asservi et constitue donc un bon exercice d'application des principes de calcul des circuits.

Le principe général de fonctionnement est illustré figure 6.2. Il consiste à commander la résistance série de l'élément de

puissance par une tension de contrôle prélevée sur la tension de sortie qui est comparée à une tension de référence fournie par exemple par une diode Zener*.

**C'est un composant particulier qui présente la propriété de fournir une tension de valeur donnée, dépendant peu du courant qui le traverse.*

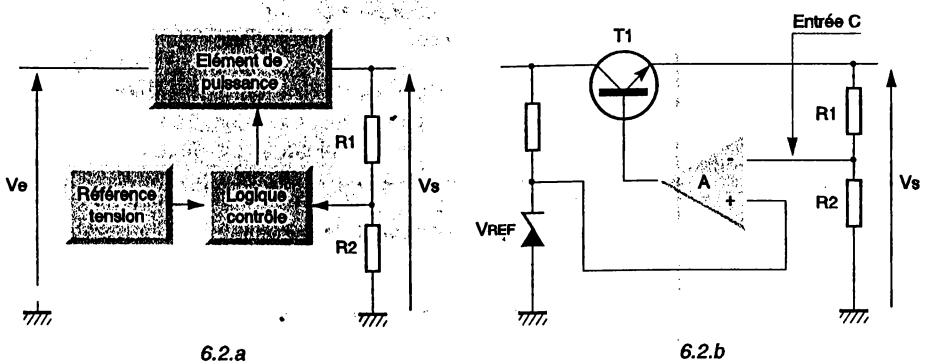



Fig. 6.2
Fonctionnement interne d'un régulateur de tension linéaire.

Si la tension de sortie est plus grande que la tension de référence, alors la résistance série augmente et la chute de tension dans cet élément de puissance augmente. La tension de sortie diminue en conséquence. Dans le schéma 6.2.b, une fraction de la tension de sortie, en l'occurrence $\frac{R_2}{R_1+R_2}\cdot V_S$, est comparée à la tension de référence $V_{REF}$ fournie par une diode Zener. L'amplificateur A modifie alors la tension de base du transistor de puissance T1. La tension d'émetteur de T1 suit celle de la base décalée de 700 mV. Il est simple de calculer la tension de sortie si on suppose le système asservi. L'amplificateur est supposé idéal. Sa tension différentielle d'entrée est alors nulle puisque son gain est infini. On écrit alors :

$$V_+ = V_-$$

et dans ce cas, on obtient :

$$\frac{R_2}{R_2+R_1}\cdot V_S = V_{REF} \quad \text{soit} \quad V_s = V_{REF}\left(1+\frac{R_2}{R_1}\right)$$

Cette relation montre qu'il est possible d'ajuster la tension de sortie en modifiant les résistances $R_1$ et $R_2$. Il faut également noter que la tension de sortie est toujours plus faible que la tension d'entrée. Dans la pratique, un écart de 2 V est la différence minimale. Les constructeurs fournissent des circuits

intégrés comprenant le transistor de puissance, l'amplificateur de régulation et la tension de référence. Il suffit donc d'ajouter les résistances de réglage et les condensateurs de filtrage selon le schéma indiqué figure 6.3.

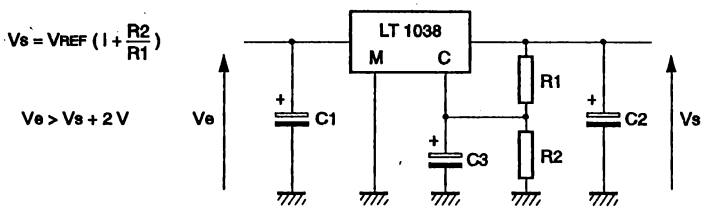



Fig. 6.3
Alimentation stabilisée pour carte.

Les capacités $C_1$, $C_2$ et $C_3$, indiquées figure 6.3, permettent d'améliorer la stabilité du montage. De plus, elles filtrent les composantes basse fréquence résiduelles (les composantes à 50 Hz par exemple). Le rejet des ondulations est alors amélioré (**ripple rejection**). Pour choisir convenablement les valeurs de ces capacités, il est conseillé de suivre les recommandations des fabricants de régulateurs. De mauvais choix peuvent en effet entraîner des instabilités et même des oscillations. Des valeurs élevées sont souvent nécessaires (10µF et au-delà). Les condensateurs au tantale ou les condensateurs chimiques sont donc utilisés.

## Exemple simple d'alimentation

Supposons maintenant que nous ne disposions pas d'une source de tension continue mais simplement d'une source de tension alternative, le 220 V alternatif par exemple. La solution la plus rapide est alors d'acheter un bloc d'alimentation fournissant la tension désirée.

Deux classes sont disponibles : les alimentations linéaires décrites précédemment et les alimentations à découpage que nous étudierons dans le paragraphe suivant. Dans certains cas cependant, par exemple pour alimenter des systèmes de petites tailles, il est nécessaire de fabriquer l'alimentation à partir de la tension alternative du secteur électrique. L'achat d'un bloc d'alimentation intégré serait trop onéreux. Le problème n'est pas très compliqué puisqu'il suffit d'ajouter, au montage de la figure 6.3, un dispositif de redressement et de filtrage. Le principe du redressement d'une tension alternative par une diode ou par un pont de diode a été étudié au chapitre 10 du premier tome. La

figure 6.4 donne un schéma type d'une telle alimentation et indique les formes des tensions aux différents points du circuit. Dans cet exemple, l'alimentation doit fournir des tensions de + 5 V et −5 V et doit pouvoir débiter 1 A sur chaque polarité. Le secteur électrique est dans un premier temps abaissé par un transformateur de puissance voulue puis redressé par un pont de diodes. La tension redressée, représentée en 6.4.c, est filtrée à l'aide de la capacité $C_1$ pour créer le signal représenté en 6.4.c appliqué en entrée du régulateur. Le signal de sortie, représenté en 6.4.d, est stabilisé en température et les composantes alternatives résiduelles sont éliminées.

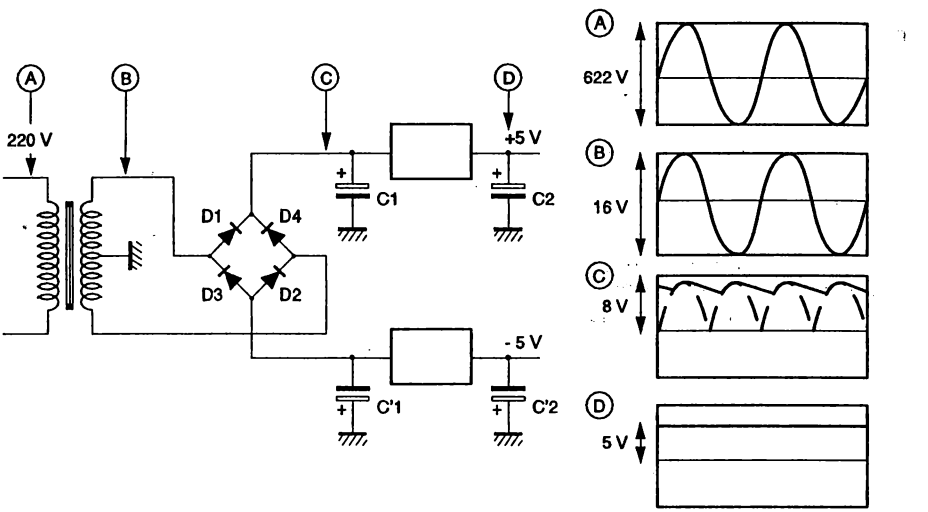



Fig. 6.4
Alimentation stabilisée à partir du secteur électrique.

Le mécanisme du filtrage par le condensateur $C_1$ est représenté sur la figure 6.5. Sa description permet de choisir la valeur de $C_1$.

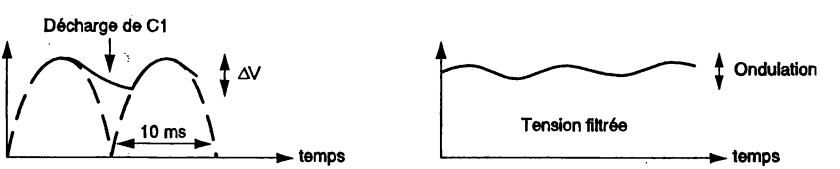



Fig. 6.5
Influence du condensateur $C_1$.

Le condensateur $C_1$ est chargé par la tension redressée puis, après la montée de la première alternance, il se décharge avec

une constante de temps donnée par l'équation des condensateurs :

$$I(t) = C_1 \cdot \frac{dV}{dt}$$

$V$ est la tension aux bornes du condensateur $C_1$.

Si on suppose le courant consommé par le circuit constant et égal à 1 A, et si on tolère, par exemple, une chute de tension au maximum égale à 1 V (une diminution plus forte entraînerait un mauvais fonctionnement du régulateur), l'équation du condensateur s'écrit pendant la demi-alternance de 10 ms (une période complète du signal alternatif est de 20 ms à 50 Hz) :

$$1A = C_1 \cdot \frac{1V}{10\ ms}$$

On en déduit facilement la valeur de $C_1$ ($C_1$ est égal à 10 000 µF). Dans cet exemple, les régulateurs sont non ajustables et deux types sont utilisés, l'un pour les tensions positives, l'autre pour les tensions négatives. Le coefficient d'abaissement de tension du transformateur est choisi pour passer de 622 V crête à crête, à 16 V crête à crête environ, ce qui permet de présenter une tension suffisante en entrée de régulateur.

## Alimentation à découpage et convertisseur continu-continu

Le régulateur étudié dans le chapitre précédent est très utilisé. Il présente cependant un certain nombre d'inconvénients :
- la tension de sortie est toujours inférieure à celle d'entrée. À partir d'une seule pile, il sera donc impossible de fournir des tensions de 5 V et plus ;
- la chute de tension $V_{CE}$ dans le transistor de puissance entraîne une dissipation de puissance dans ce transistor égale à $V_{CE}I_s$.

$I_s$ est le courant débité par l'alimentation. Cette puissance dissipée inutilement dans le transistor peut représenter plus de la moitié de la puissance totale fournie par le secteur électrique. Le rendement en puissance est donc assez mauvais.

Ces deux raisons ont amené à développer un autre type de régulateur : le **régulateur à découpage**.

La figure 6.6 illustre le fonctionnement d'un tel dispositif.

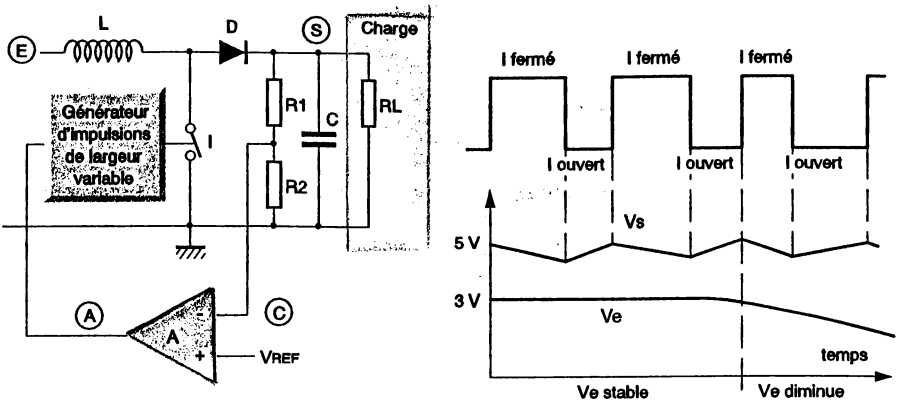



Fig. 6.6
Fonctionnement interne d'un régulateur à découpage.

La structure générale est identique à celle du régulateur linéaire. La tension de sortie est prélevée puis comparée à une tension de référence $V_{REF}$. Dans ce cas, la tension de commande ne modifie pas la résistance série d'un élément de puissance mais dérive le courant fourni pendant une fraction du temps. Cette fraction est déterminée par un générateur d'impulsions de durées réglables, commandé par la tension de sortie de l'amplificateur opérationnel. Ce générateur commande un interrupteur de puissance, généralement un transistor, qui dérive tout simplement le courant à la masse quand il est commandé. L'inductance *L* en série dans l'entrée permet de créer une surtension à ses bornes et donc d'élever la tension d'entrée. Ce montage élève la tension continue d'entrée et réduit la perte de puissance dans l'élément de dissipation. Quand l'interrupteur de puissance est fermé, la chute de tension à ses bornes est quasi-nulle et une puissance faible est perdue. Une analyse détaillée permettra au lecteur de comprendre en détail le fonctionnement de ce circuit.

Le but de ce calcul est de préciser les valeurs des tensions et des courants et montrer que la valeur moyenne de la tension de sortie peut être plus élevée que la tension d'entrée. Pour cela, il est utile de reprendre la figure 6.6 en la simplifiant. Le circuit de régulation proprement dit, formé du pont de résistances et de l'amplificateur opérationnel, peut être omis puisqu'il constitue une charge à haute impédance pour le circuit. Le dispositif de puissance sera assimilé à un court-circuit ou à un circuit ouvert, en fonction du cycle de fonctionnement.

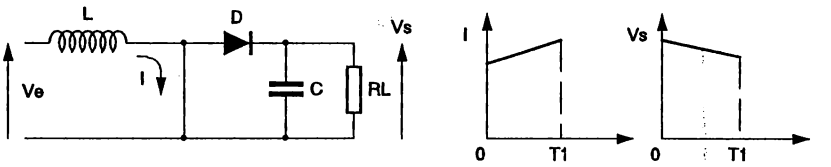



Fig. 6.7.a
Période de 0 à $T_1$. Interrupteur I fermé.

La figure 6.7.a représente le premier cycle de fonctionnement. On suppose que la tension de sortie $V_s$ est supérieure à la tension d'entrée $V_e$. Cette hypothèse sera bien sûr vérifiée. On suppose également, que dans cette phase, l'interrupteur de puissance est fermé. Cette phase dure de 0 à $T_1$. La diode D est alors polarisée en inverse et elle est donc non passante. Le condensateur C se décharge dans la résistance $R_L$ avec une constante de temps RC. Si cette valeur est importante, la valeur de la tension de sortie peut varier faiblement. Le courant $I(t)$ est donné par la relation :

$$V_e(t) = L \cdot \frac{dI}{dt}$$

Il augmente donc à partir de sa valeur initiale avec une pente constante si on suppose la tension d'entrée $V_e$ quasi-constante :

$$I(T_1) = I(0) + \frac{V_e \cdot T_1}{L}$$

À partir de $T_1$, l'interrupteur de puissance s'ouvre et le schéma se modifie comme indiqué en 6.7.b.

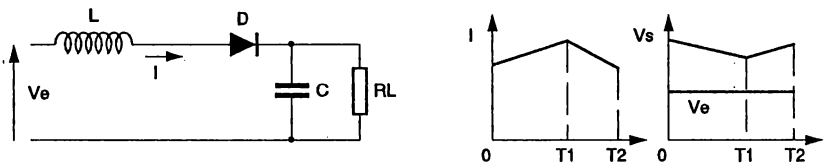



Fig. 6.7.b
Période de $T_1$ à $T_2$. Interrupteur I ouvert.

Reprenons un résultat important du chapitre 2 : le courant traversant une self ne peut varier brusquement. Dans notre exemple, le courant traversant l'inductance a donc la valeur initiale $I(T_1)$, ce qui entraîne la mise en conduction de la diode D. On suppose que le système a atteint un régime d'équilibre. Nécessairement, les valeurs de $I$ et de $V_s$ au temps $T_2$ seront

égales aux valeurs qu'elles avaient en $t$ égal à 0. Le circuit fonctionne en effet selon un cycle régulier et les valeurs d'équilibre sont nécessairement périodiques quand le régime d'équilibre est atteint.

Le courant $I$ va donc décroitre à partir de $T_1$, ce qui veut dire que sa dérivée est négative. $V_s$ revient à sa valeur initiale. La tension $V_S$ s'écrit alors :

$$V_S(t) = V_e - L \cdot \frac{dI}{dt} - V_D$$

$V_D$ est la tension aux bornes de la diode, sa valeur est d'environ 700 mV. Comme la dérivée du courant est négative, la tension $V_S$ est supérieure à $V_e$, ce qui est en accord avec l'hypothèse initiale. Le circuit a donc élevé la tension d'entrée.

Que se passe-t-il quand la tension d'entrée varie et comment mettre en évidence la fonction de régulation du circuit ? Si par exemple $V_e$ diminue, alors l'amplificateur A de régulation détecte la décroissance. Il modifie ensuite la valeur de $T_1$ de telle manière que la diminution de $V_s$ soit plus faible. La pente de décroissance donnée par le produit $R_L C$ est en effet constante. La sortie est donc compensée.

Le principe du découpage a donc permis de fournir une tension de sortie de valeur plus élevée que celle d'entrée. De plus, l'élément de puissance, quand il est mis en conduction, ne dissipe que peu de puissance, le rendement du régulateur est donc excellent. Pourquoi dans ce cas les régulateurs de ce type n'ont-ils pas totalement remplacé les régulateurs linéaires ?

Pour répondre à cette question, il faut revenir sur les formes d'ondes et comprendre que, par principe, ces régulateurs introduisent une fréquence indésirable, la fréquence de découpage.

Aussi parfaits soient les filtres de réjection, une composante résiduelle de cette fréquence et de ses harmoniques est présente. Pour certaines applications critiques, principalement dans le domaine analogique, ces composantes peuvent être la cause d'un mauvais fonctionnement. Dans ce cas, le régulateur linéaire retrouve sa place. Les fréquences de découpage sont maintenant de l'ordre de plusieurs centaines de kHz.

Les techniques permettant de se protéger contre les ondulations résiduelles ont été exposées au chapitre 3 (familles logiques). Elles font appel aux découplages des circuits, à la conception des plans de masse, à l'utilisation des filtres secteurs et aux techniques de blindage des éléments rayonnants ou des éléments sensibles.

# Caractéristiques des régulateurs et critères de choix

Quelques produits du commerce sont pris comme exemple pour illustrer les considérations précédentes.

## 1 Régulateurs linéaires

| | LT 1038 | LM 345 | LM 317 |
|---|---|---|---|
| Polarité | positive | négative | positive |
| Tension sortie | 1,2 V à 32 V | – 5 V | 1,2 V à 37 V |
| Tension entrée min | $V_s$ + 3 V | – 7,5 V | $V_s$ + 3 V |
| Courant max sortie | 10 A | 3 A | 0.1 A |
| Protection sortie* | oui | oui | oui |
| Rejet ondulations** | 75 dB | 65 dB | 65 dB |
| Stabilité température | 1 % | 1 % | 0,5 % |

* : Protection contre un court-circuit éventuel,

** : Exprime le rapport entre une ondulation d'entrée et celle en sortie à 50Hz et pour des conditions données de découplage.

## 2 Régulateurs à découpage

| | LT1044 | LT1172 |
|---|---|---|
| Polarité | positive et négative | positive et négative |
| Puissance sortie max | | 20 W |
| Tension entrée | $V_s$/2 environ | $V_s$/2 environ |
| Courant sortie max | 100 mA | |
| Rendement | 95 % | 95 % |
| Fréquence découpage | Variable | 100 kHz |

Chapitre 7

# LES CONVERSIONS ANALOGIQUE-NUMÉRIQUE ET NUMÉRIQUE-ANALOGIQUE

## SOMMAIRE


- Conversion analogique-numérique et acquisition de données
- Choix de la fréquence d'échantillonnage
- Choix du nombre de bits de codage
- Convertisseur parallèle
- Convertisseur à approximations successives
- Convertisseur à rampe
- Convertisseur sigma-delta
- Récapitulatif des convertisseurs analogique-numérique
- Convertisseur numérique-analogique
- Interfaces avec les microprocesseurs
- Caractéristiques des convertisseurs et critères de choix
- Entraînement


*Les techniques actuelles font de plus en plus appel au traitement numérique qui présente d'intéressantes propriétés de performance et de souplesse. La généralisation de l'utilisation des microprocesseurs a rendu ces techniques réellement applicables à de nombreux cas à partir des années 1980. Le domaine des télécommunications est sans doute celui dans lequel l'application des techniques numériques est la plus développée (numérisation de la voix pour la téléphonie, généralisation du fax et du courrier électronique, enregistrement numérique des disques, etc.). La commande des machines est également un secteur où les techniques numériques connaissent un vif succès (commande de moteurs, régulation de chauffage, etc.). Dans toutes ces applications, il est cependant nécessaire de faire la liaison entre le monde dans lequel nous vivons qui est analogique, et le monde du traitement qui est numérique. C'est pourquoi l'industrie électronique a développé sous forme de circuits intégrés des convertisseurs extrêmement performants et de faible coût pour réaliser ces opérations.*
*La figure 7.1, page suivante, illustre l'intérêt de ces circuits intégrés spéciaux.*

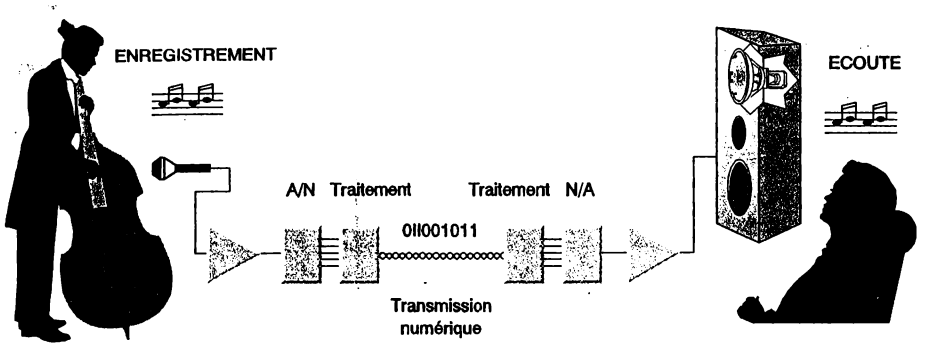



Fig. 7.1
Conversion analogique-numérique et numérique-analogique.

## Conversion analogique-numérique et acquisition de données

Comme il a été dit dans l'introduction générale, il est nécessaire, dans de nombreuses applications, de transformer une valeur continue mesurée sur un signal en une valeur numérique codée en binaire.

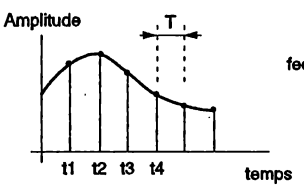
Les mesures de température, les mesures de débit ou de positions sont par exemple traitées par les logiciels de contrôle des procédés industriels. Dans un autre secteur, voix et musique sont enregistrées sous forme numérique. De nombreux développements s'effectuent également pour traiter l'image. Il est alors nécessaire de diviser l'image en un nombre important de petits carrés élémentaires appelés pixels. Chaque pixel est analysé dans ses trois composantes de couleur fondamentales, et l'information correspondante, c'est-à-dire la donnée des intensités des trois composantes, est numérisée pour un traitement ultérieur.

L'application donnée chapitre 12 illustre cette technique de manière précise. Dans tous les cas, il est nécessaire de mesurer le signal à un temps donné puis de transformer cette valeur continue mesurée en une valeur codée en binaire.

Les instants de mesure ou instants d'échantillonnage sont généralement répartis uniformément. On définit ainsi une **fréquence d'échantillonnage** comme l'indique la figure 7.2. Deux questions se posent alors immédiatement : comment choisir la fréquence d'échantillonnage ? Comment choisir le nombre de bits de codage ?

La réponse à ces deux questions de base constitue l'essentiel des critères de conception d'une chaîne de traitement numérique de données. Le problème pratique est plus complexe puisque les

différents signaux à traiter ne demandent pas nécessairement la même fréquence d'échantillonnage. Prenons par exemple le cas d'une automobile. Il est utile de mesurer la température du liquide de refroidissement à une fréquence relativement lente (une fois par seconde par exemple). Pour contrôler l'allumage ou le freinage, une fréquence plus élevée est indispensable. Le système d'acquisition des données doit donc prendre en compte plusieurs voies de mesure échantillonnées à des fréquences différentes. Après avoir codé les différents échantillons, il transporte les valeurs binaires vers un processeur de traitement ou vers une mémoire numérique de stockage. Dans la pratique, il est souvent commode d'utiliser des lignes électriques réservées aux transferts de données dans le système. Ces lignes sont appelées **bus de communication**.




$$fech = \frac{1}{T}$$

| Temps | Valeur codée | Echantillon |
|---|---|---|
| t1 | 0 1 1 0 | Echantillon 1 |
| t2 | 0 1 1 1 | Echantillon 2 |
| t3 | 0 1 1 0 | Echantillon 3 |
| t4 | 0 1 0 0 | Echantillon 4 |

Fig. 7.2
Échantillonnage et conversion analogique-numérique.

La figure 7.3 donne une représentation schématique d'un tel système. Une solution simple consiste à associer, à chaque grandeur à mesurer, un convertisseur analogique-numérique. Cette solution est cependant peu économique et réservée à l'échantillonnage des grandeurs demandant une fréquence élevée et se prêtant mal à un partage du circuit de conversion.

Dans la plupart des applications, plusieurs grandeurs à mesurer sont regroupées sur un même convertisseur selon l'architecture indiquée figure 7.3. Les signaux, généralement de faible valeur, sont amplifiés, filtrés puis appliqués à un multiplexeur analogique dont la fonction est de choisir une voie parmi *n*. Dans la pratique, les signaux sont appliqués à une batterie de *n* interrupteurs commandés par une logique d'adressage permettant de choisir l'un d'entre eux. Des transistors rapides, bloqués ou passants, peuvent faire office d'interrupteurs. Ils doivent avoir une vitesse de commutation suffisante. Le choix de la voie à mesurer à un instant donné est effectué par un circuit logique spécial appelé séquenceur qui, tel un chef d'orchestre, décide des différentes phases de fonctionnement du système. Il choisit également le début de la conversion puis le transfert des bits de conversion vers

le bus de communication. Il peut également piloter l'entrée des données dans le microprocesseur de traitement ou vers la mémoire numérique. Quand la conversion est achevée sur une voie, il décide alors de passer à la suivante et le même cycle s'effectue.

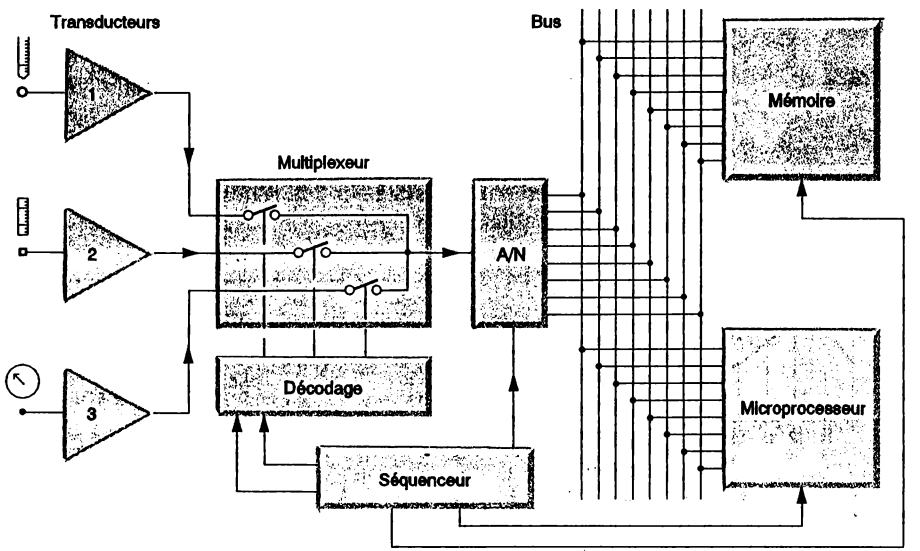



Fig. 7.3
Système d'acquisition des données.

Dans la pratique, le microprocesseur peut assurer toutes les fonctions de découpage des opérations dans le temps en déroulant un programme qui génère des impulsions logiques de contrôle.

La figure 7.4 représente le déroulement temporel des opérations. Dans cet exemple simple, deux grandeurs sont échantillonnées à des fréquences différentes et les bits de codage sont présentés sur un bus de données puis inscrits dans une mémoire ou entrés dans un microprocesseur. Le signal 1 variant plus rapidement que le signal 2, il est échantillonné à une fréquence plus élevée afin de ne pas perdre d'informations. Le choix de la voie à échantillonner s'effectue à l'aide du multiplexeur analogique. Le numéro de la voie est lu en sortie du séquenceur puis décodé pour activer électriquement l'interrupteur sélectionné. Après un court intervalle de temps, correspondant à la stabilisation des données en sortie du multiplexeur, la conversion peut démarrer. Elle dure un temps plus ou moins long dépendant du convertisseur choisi. Ce temps est appelé **temps de conversion**. À la fin de ce temps de

conversion, les données binaires sont prêtes en sortie du convertisseur. Les circuits utilisateurs (microprocesseur et mémoire) sont connectés en parallèle sur le bus de données comme l'indique la figure 7.3. L'application donnée chapitre 12 détaille un cas où les échantillons sont écrits dans une mémoire rapide. Dans d'autres applications, les bits correspondant aux divers échantillons sont présentés en entrée du microprocesseur et traités au fur et à mesure.

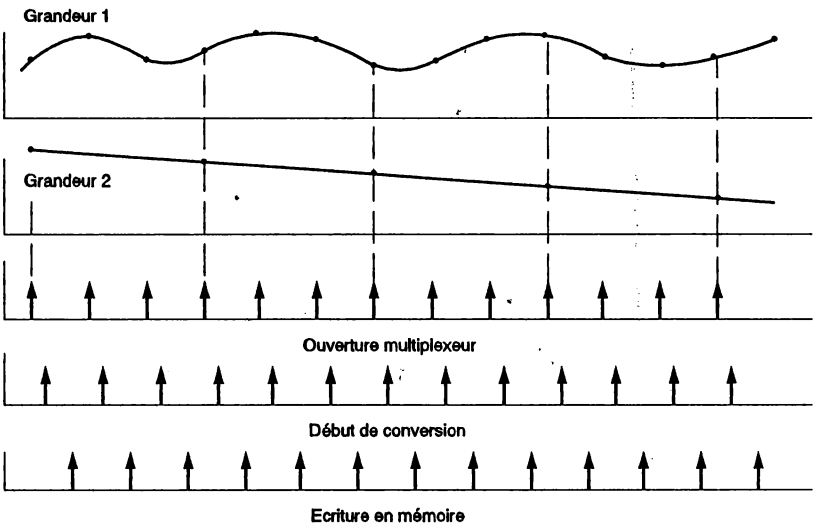



Fig. 7.4
Déroulement temporel des opérations.

Revenons maintenant aux deux questions de base : comment choisir la fréquence d'échantillonnage et comment déterminer le nombre de bits de codage ?

## Choix de la fréquence d'échantillonnage

Déjà évoqué au chapitre 7 du premier tome, ce choix sera rappelé dans ses grandes lignes et complété par une annexe mathématique. Il découle du célèbre **théorème de Shannon** : pour conserver toute l'information contenue dans un signal, il faut l'échantillonner à une fréquence plus élevée que deux fois la fréquence la plus haute de son **spectre**. Le spectre d'un signal a été défini au chapitre 5 du premier tome. C'est, rappelons-le,

l'ensemble de toutes les sinusoïdes dont la somme constitue le signal. La règle de Shannon est illustrée figure 7.5.a. Enoncée aussi simplement, cette règle conduit cependant à se poser un certain nombre de questions :
- Que veut dire exactement conserver l'information ?
- Que se passe-t-il quand le spectre du signal n'est pas borné ?
- Est-il nécessaire de filtrer avant l'échantillonnage ?

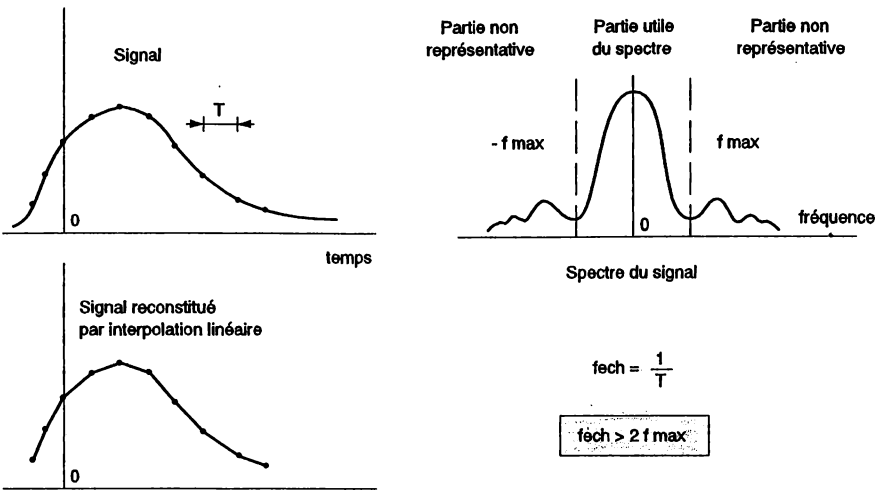



Fig. 7.5.a
Règle de Shannon.

Conserver toute l'information veut dire qu'il est possible, à partir des échantillons, de reconstituer un signal strictement identique au signal d'origine. La figure 7.5.a illustre une reconstitution possible obtenue par une simple interpolation entre les valeurs des échantillons. Il faut noter au passage que cette reconstitution n'est pas la meilleure et que l'interpolation optimale est plus compliquée. Elle seule permet une reconstitution parfaite du signal.

Le spectre du signal n'est généralement pas borné. La figure 7.5.a donne une représentation graphique des composantes marginales au-delà de la fréquence $f_{max}$ identifiée comme la fréquence de coupure du spectre. Ces composantes sont par exemple des résidus de filtrage ou des composantes de bruit électronique. La définition de $f_{max}$ peut donc avoir un caractère assez subjectif si la différence n'est pas nette entre les fréquences principales du signal et ces composantes marginales.

Filtrer avant l'échantillonnage à l'aide d'un filtre à flancs très raides permet de se placer dans les conditions d'applications du théorème de Shannon. Un tel filtre est appelé **filtre anti-repliement** (*aliasing filter*). Il ne faudrait pas en déduire hâtivement qu'il est alors possible de choisir une valeur quelconque pour la fréquence d'échantillonnage puisqu'il suffit de choisir la fréquence de coupure du filtre égale à la moitié de cette fréquence. En effet, si cette fréquence de coupure est inférieure à la fréquence la plus haute de la partie significative du spectre, une grande partie de l'information contenue dans le signal est perdue. En définitive et dans tous les cas, la fréquence d'échantillonnage doit être supérieure à deux fois la fréquence la plus haute de la partie significative du spectre. Dans la pratique, un facteur trois est souvent choisi. Le filtre anti-repliement permet d'éliminer les composantes indésirables à plus haute fréquence. Son utilisation est fortement conseillée dans les applications.

Pour terminer cet important paragraphe, il est possible de donner quelques valeurs numériques. Un signal a par exemple un spectre compris entre 0 et 20 kHz. Une fréquence d'échantillonnage au minimum égale à 40 kHz est donc choisie. Dans la pratique, une valeur de 60 kHz est conseillée.

Le paragraphe qui suit donne une démonstration du théorème d'échantillonnage. Le lecteur peu familier avec les notations mathématiques pourra sans inconvénient passer au paragraphe suivant.

Le signal $x(t)$ a une transformée de Fourier $X(f)$ qui par hypothèse a des valeurs non nulles dans un domaine fini comme l'indique la figure 7.5.b. Cette hypothèse de départ est la condition d'application du théorème.

Dans un souci de simplification, la figure ne représente que la partie réelle de la transformée. La partie imaginaire n'est pas représentée. L'idée générale de la démonstration est de construire à partir de $X(f)$ un signal périodique $X_p(f)$ dans le domaine fréquentiel. Ce signal est alors décomposable en série de Fourier, c'est-à-dire en une série discrète de fonctions harmoniques. Il suffit ensuite de passer de $X_p(f)$ à $X(f)$.

La figure 7.5.b indique la construction de $X_p(f)$. Le signal $X(f)$ est répété avec une période $f_e$ choisie de telle manière que les éléments répétés ne se chevauchent pas. Cela serait impossible si le spectre n'était pas limité. L'examen de la figure montre qu'il est nécessaire que $f_e$ soit supérieure à $2f_{max}$ pour éviter les recouvrements de spectre.

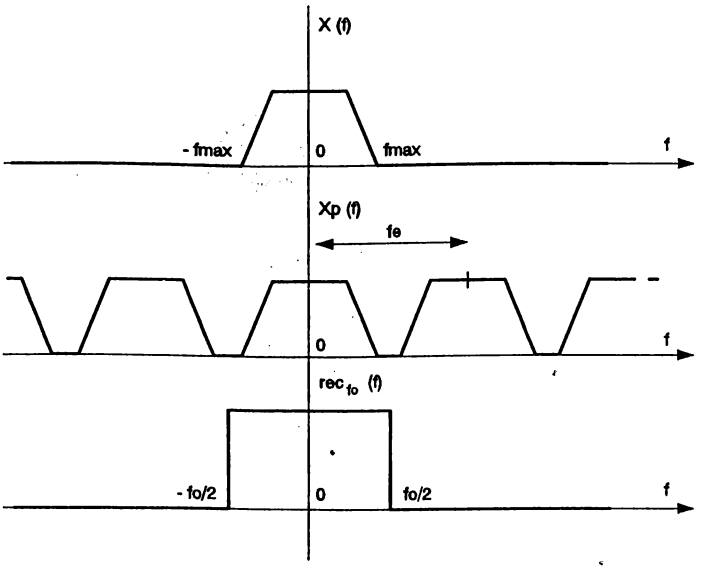



Fig. 7.5.b
Règle de Shannon (démonstration).

On passe simplement de $X_p(f)$ à $X(f)$ par l'introduction de la fonction rectangle $\text{Rec}_{f_0}$ représentée sur la figure. La valeur $f_0$ est choisie entre $f_{max}$ et $f_e - f_{max}$ :

$$X(f) = X_p(f) \cdot \text{Rec}_{f_0}(f)$$

$X_p(f)$ se décompose en série de Fourier puisqu'elle est périodique :

$$X_p(f) = \sum_{n=-\infty}^{n=+\infty} a_n \cdot \exp^{j2\pi n \frac{f}{f_e}}$$

avec :

$$a_n = \frac{1}{f_e} \times \int_{-f_{max}}^{+f_{max}} X(f) \cdot \exp^{-j2\pi n \frac{f}{f_e}} \cdot df$$

Il est facile d'identifier $a_n$ avec la valeur de $x$ au temps $t$ égal à $-n/f_e$. C'est la définition de la transformée de Fourier. Si on note $T_e$ (égal à $1/f_e$) la période, on obtient :

$$X_p(f) = \sum_{n=-\infty}^{n=+\infty} T_e \cdot x(-nT_e) \exp^{j2\pi n T_e f}$$

Il est maintenant facile d'exprimer $X(f)$ en fonction de $X_p(f)$. Il suffira ensuite de prendre la transformée de Fourier inverse pour obtenir la décomposition de $x(t)$ en série discrète :

$$x(t) = F^{-1}\left\{\sum_{n=-\infty}^{n=+\infty} T_e x(-nT_e)\cdot \exp^{j2\pi nT_e f}\cdot Rec_{f_0}(f)\right\}$$

Si on applique le théorème de translation (multiplication par un terme exponentiel) et si on utilise le fait que la transformée de Fourier de la fonction rectangle est un sinus cardinal, on obtient :

$$x(t) = \sum_{n=-\infty}^{n=+\infty} x(nT_e)\cdot \mathrm{sinc}(f_0[t-nT_e])$$

Pour cette opération, l'indice $n$ a été changé en $-n$ pour exprimer $x(nT_e)$ et non pas $x(-nT_e)$.

La fonction sinc est définie, rappelons-le, par :

$$\mathrm{sinc}(x) = \frac{\sin \pi x}{\pi x}$$

En choisissant la valeur $f_0$ égale à $f_e$, on obtient :

$$x(t) = \sum_{n=-\infty}^{n=+\infty} x(nT_e)\cdot \mathrm{sinc}\left(\frac{t}{T_e} - n\right)$$

Cette importante formule permet d'exprimer le signal initial en fonction des valeurs prises aux instants d'échantillonnage et d'elles seules. La condition $f_e$ supérieure à $2f_{max}$ se déduit de la construction de la fonction périodique. Si cette condition n'était pas remplie, il serait impossible d'écrire la relation :

$$X(f) = X_p(f)Rec_{f_0}(f)$$

La valeur du signal à un instant $t$ n'est pas donnée par une simple interpolation entre les valeurs prises aux temps $nT_e$ et $(n+1)T_e$. Elle est donnée par une série infinie de termes, comme l'indique la formule ci-dessus. Dans la pratique, on limite le calcul à une série finie de termes, ce qui introduit une erreur dite de troncature.

## Choix du nombre de bits de codage

Il s'agit donc de coder chaque échantillon avec un nombre de bits suffisant pour conserver une bonne précision. Un codage sur 8 bits permettra par exemple de placer l'amplitude du signal parmi 256 valeurs possibles et d'avoir ainsi une précision de l'ordre de 1/256 quand le signal a la valeur maximale. On comprend alors l'intérêt de choisir un nombre important de bits de codage dans les mesures à haute précision. Quand le signal a une amplitude plus ou moins constante, le choix est assez simple. Supposons par

exemple un signal d'amplitude moyenne 100 mV et un convertisseur 8 bits acceptant des signaux de 0 à 10 V. Il suffit de placer un amplificateur de gain 50 pour avoir un signal de 5 V environ en entrée du convertisseur et être donc dans une plage intéressante. Un gain plus élevé serait dangereux puisque la valeur pourrait être supérieure à la valeur maximale acceptable par le convertisseur. Une valeur plus faible serait peu intéressante puisque l'erreur due au nombre limité de bits de codage est d'autant plus gênante que la valeur codée est faible. Donnons un exemple simple : une valeur codée au quart de la pleine échelle, soit 2.5 V, est affectée d'une erreur de l'ordre de 1/64 puisque 2.5 V est codé au canal 64.

Etudions maintenant le cas d'un signal d'entrée variant dans une large gamme. Le phénomène s'aggrave quand la plage de variation du signal augmente (**dynamique**). Si la dynamique du signal précédent est de 10, le signal maximum est supposé de valeur 5 V en entrée de codeur. Une valeur de 500 mV est maintenant possible. Le canal de codage correspondant est 128/10 soit 13 environ. L'erreur relative est maintenant de l'ordre de 1/13, ce qui est loin d'être négligeable. La dynamique du signal entraîne donc une augmentation du nombre de bits de codage quand il est nécessaire de conserver un bonne précision dans toute la gamme de codage.

**Dynamique du signal**
**⇒ Choix du nombre de bits**
**Précision demandée**

La figure 7.6 illustre de manière plus précise l'erreur due au nombre limité de bits de conversion. Cette erreur est appelée **erreur de quantification**. Un codage sur trois bits est représenté dans un souci de simplification mais les résultats sont transposables à un nombre de bits quelconque. En abscisse est représentée la valeur de la tension à coder, et en ordonnée le code binaire correspondant. Deux types de codage sont représentés : le premier (figure 7.6) introduit une erreur systématique, le second (figure 7.7) est dit centré. Dans le codage sur trois bits, les huit niveaux possibles correspondent à des variations de 10/8 V soit 1.25 V, quand on passe de l'un à l'autre. Cette valeur est le quantum correspondant au nombre de bits de codage. Elle est appelée **LSB** (*least signifiant bit*). Dans la figure 7.6 la valeur initiale codée est 000 quand le signal d'entrée est nul. Quand il franchit la valeur 1.25 V, la valeur codée devient 001, puis 010 quand la valeur 2.5 V est atteinte et ainsi de suite jusqu'à la valeur 10 V.

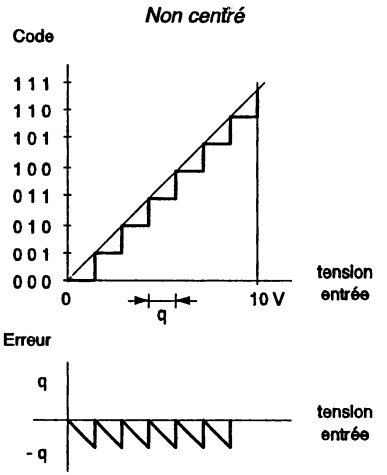



Fig. 7.6
Codage et erreur de quantification (non centré).

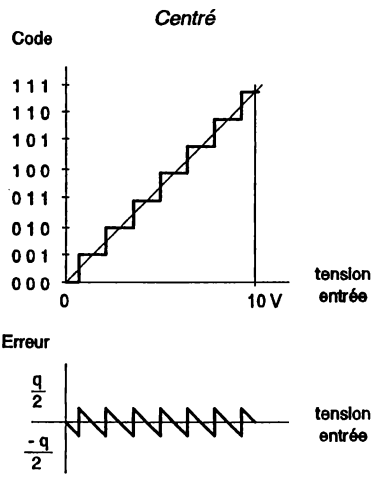



Fig. 7.7
Codage et erreur de quantification (centré).

L'**erreur de quantification** est la différence entre la valeur appliquée en entrée du convertisseur et la valeur correspondant exactement au code obtenu en sortie du convertisseur. Elle est évidemment nulle quand la valeur du signal est égale à l'une des valeurs de basculement d'un code à l'autre. La valeur de cette erreur est représentée sur le graphe de la figure 7.6. Dans le codage de la figure 7.6, elle est toujours négative puisque la valeur codée correspond à une tension toujours inférieure à la valeur réellement appliquée. Elle augmente linéairement tant que le signal n'a pas atteint la valeur de basculement. Le codage 7.6 introduit donc une erreur systématique dont la valeur moyenne est négative. Pour remédier à cela, le codage représenté figure 7.7, effectivement utilisé dans les produits du commerce, est dit centré puisque les seuils de basculement des codes sont décalés d'un demi quantum par rapport au codage initial 7.6. Le premier seuil de basculement est par exemple 0.625 V et non pas 1.25 V. Il est facile de constater dans ce cas que l'erreur de quantification représentée en fonction du signal à coder varie comme l'indique le graphe de la figure 7.7. Elle est simplement décalée d'un demi quantum par rapport à l'erreur du codage 7.6. Pour caractériser cette erreur, il faut utiliser les grandeurs statistiques traditionnelles : l'espérance mathématique ou moyenne et la variance ou fluctuation moyenne. Ces deux grandeurs ont été introduites au chapitre 7 du premier tome pour l'étude des effets

du bruit électronique. Le calcul de la variance du bruit de quantification est donné ci-dessous. La démonstration peut être négligée dans une première lecture.

On se place donc dans le cas décrit figure 7. Le codage est centré. Les calculs statistiques s'effectuent en considérant toutes les valeurs possibles de la tension d'entrée. Pour simplifier le calcul, elles sont supposées équiprobables. La densité de probabilité est donc dans ce cas :

$$p(V) = \frac{1}{V_{MAX}} \mathrm{d}V$$

On vérifie bien :

$$\int_0^{V_{MAX}} p(V)\mathrm{d}V = 1$$

Le quantum $q$ est défini de la manière suivante, si $n$ est le nombre de bits de codage :

$$q = \frac{V_{MAX}}{2^n}$$

Le calcul de la moyenne de l'erreur $E(e)$ est immédiat. Les valeurs positives et négatives étant également réparties et la densité de probabilité étant constante, la moyenne statistique de l'erreur est alors nulle.

Le calcul de la variance est simple si on tient compte du caractère périodique de cette erreur.

Il suffit de la calculer sur une période en appliquant la relation :

$$Var(e) = \int_0^{V_{MAX}} p(V)e(V)^2\mathrm{d}V = 2^n \int_{q/2}^{3q/2} p(V)(q-V)^2\mathrm{d}V$$

La période choisie va de $q/2$ à $3q/2$.

Le calcul conduit alors au résultat suivant :

$$Var(e) = \frac{1}{q}\int_{q/2}^{3q/2}(q-V)^2\mathrm{d}V = \frac{q^2}{12}$$

La racine carrée de la variance est appelée écart-type $\sigma$ égal à :

$$\sigma = \frac{q}{\sqrt{12}}$$

Cette mesure de l'erreur de quantification est souvent exprimée en terme de rapport signal sur bruit. Le bruit est mesuré par l'écart-type calculé ci-dessus alors que le signal est mesuré par sa valeur efficace. Pour un signal sinusoïdal de valeur crête à crête

égale à la valeur maximale acceptée par le convertisseur, la valeur efficace est alors $\frac{V_{max}}{2 \cdot \sqrt{2}}$ et le rapport signal sur bruit est donc :

$$\frac{12}{8} \cdot 2^{2n}.$$

Exprimé en décibels, on obtient :

$$\xi^2 \text{dB} = 10 \log_{10} \xi^2$$

soit

$$\xi^2 \text{dB} = 1{,}77 + 6{,}02 \cdot n$$

Quelques valeurs typiques précisent l'influence du bruit de quantification :

$n = 4$ $\quad \zeta^2 = 26$ dB
$n = 8$ $\quad \zeta^2 = 50$ dB
$n = 16$ $\quad \zeta^2 = 98$ dB

## Convertisseur parallèle

Le **convertisseur parallèle** (*flash converter*) est basé sur un principe de fonctionnement simple. Ce type de convertisseur est de plus en plus utilisé dans les applications demandant une fréquence d'échantillonnage élevée puisque son temps de conversion peut être aussi faible que 10 nanosecondes (une nanoseconde est un milliardième de seconde). Le traitement de l'image et de la voix, les circuits de télécommunication et les régulations rapides utilisent largement ce type de convertisseur. La figure 7.8 illustre le principe de fonctionnement du convertisseur parallèle dans le cas d'une conversion sur trois bits. Le cas est facilement généralisable à un nombre de bits quelconque.

La tension à coder $V_e$ est appliquée en parallèle sur les sept comparateurs (voir chapitre 11 tome 1) dont les seuils de comparaison sont déterminés à l'aide d'un pont de résistances. Les valeurs de basculement sont $V_{REF}/16$, $3V_{REF}/16$, $5V_{REF}/16$, $7V_{REF}/16$, $9_{REF}/16$, $11V_{REF}/16$, $13V_{REF}/16$. Quand la tension *Ve* dépasse un seuil, le comparateur concerné voit sa sortie passer de l'état bas à l'état haut. Au fur et à mesure que la tension augmente, les comparateurs déclenchent, les uns derrière les autres. Les sorties logiques des sept comparateurs représentent donc la valeur du signal $V_e$ qu'il faut coder. Il ne faudrait pas en conclure hâtivement que les sept niveaux logiques en sortie des comparateurs représentent la valeur codée en binaire du signal d'entrée.

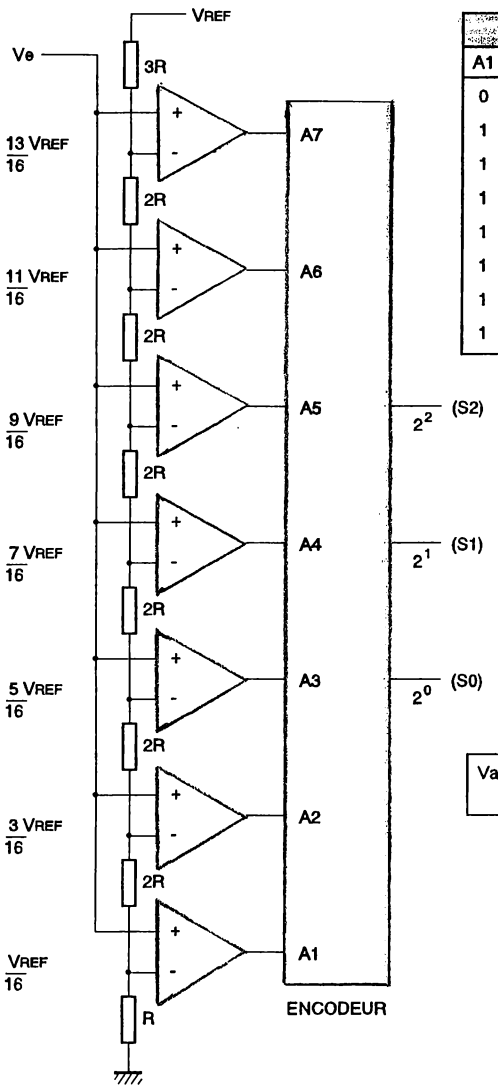



**Table de vérité de l'encodeur**

| A1 | A2 | A3 | A4 | A5 | A6 | A7 | S2 | S1 | S0 |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

Valeur binaire : $S2.2^2 + S1.2^1 + S0.2^0$
avec si = 0 ou 1

Fig. 7.8
Convertisseur analogique-numérique parallèle.

Le tableau de la figure 7.8 montre le transcodage qu'il est nécessaire d'effectuer pour obtenir la valeur codée sur trois bits. Il est possible d'écrire les équations logiques reliant les sorties $A_i$ des comparateurs aux bits de codage $S_i$ :

$$S_0 = A_1\overline{A}_2\overline{A}_3\overline{A}_4\overline{A}_5\overline{A}_6\overline{A}_7 + A_1A_2A_3\overline{A}_4\overline{A}_5\overline{A}_6\overline{A}_7 + A_1A_2A_3A_4A_5\overline{A}_6\overline{A}_7 + A_1A_2A_3A_4A_5A_6A_7$$

$$S_1 = A_1A_2\overline{A}_3\overline{A}_4\overline{A}_5\overline{A}_6\overline{A}_7 + A_1A_2A_3\overline{A}_4\overline{A}_5\overline{A}_6\overline{A}_7 + A_1A_2A_3A_4A_5\overline{A}_6\overline{A}_7 + A_1A_2A_3A_4A_5A_6A_7$$

$$S_2 = A_1A_2A_3A_4\overline{A}_5\overline{A}_6\overline{A}_7 + A_1A_2A_3A_4A_5\overline{A}_6\overline{A}_7 + A_1A_2A_3A_4A_5A_6\overline{A}_7 + A_1A_2A_3A_4A_5A_6A_7$$

Ces équations logiques se déduisent simplement du tableau de transcodage indiqué figure 7.8. Pour écrire ces équations logiques, il suffit de considérer tous les cas où un bit $S_i$ donné prend la valeur « 1 » et d'écrire le OU logique des combinaisons des bits $A_i$ correspondants. Ces équations peuvent se simplifier par les techniques classiques de réduction des expressions logiques. Le cas trois bits considéré est proposé au lecteur à titre d'exercice. Il est ensuite possible de procéder à la synthèse logique de ces expressions en associant les portes correspondantes.

Les bits de codage des échantillons ayant été obtenus, ils sont ensuite inscrits dans une mémoire ou appliqués en entrée d'un microprocesseur. Un signal de commande permet de définir le temps de mesure de l'échantillon et le début de la conversion. La conversion dure le temps qu'il faut aux différents signaux pour se propager à travers les différents circuits. Comme toutes les opérations s'effectuent en parallèle, ce temps est court. Dans l'exemple donné figure 7.9, le début de la conversion est défini par le front montant de l'impulsion *CK* de commande. Les bits de codage sont valides au moment du front montant suivant. La conversion et la mise à disposition des bits de codage s'effectuent donc au rythme de l'horloge d'échantillonnage. Il y a un retard d'une période d'horloge entre le temps d'échantillonnage et la fourniture des bits de codage.

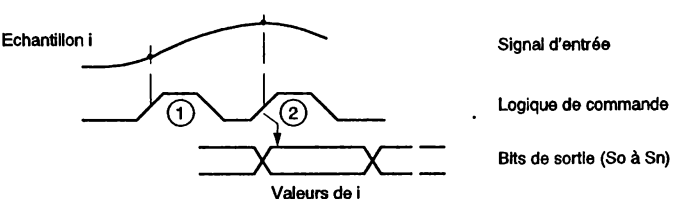



Fig. 7.9
Convertisseur parallèle, chronogramme.

Les fréquences d'échantillonnage peuvent avoir des valeurs aussi élevées que 500 MHz. Dans les applications courantes, des convertisseurs 8 bits fonctionnant à 100 MHz sont disponibles sous forme de circuits intégrés. Quand le nombre de bits augmente, des difficultés importantes apparaissent. Dans le cas d'un codage sur 10 bits par exemple, il faut imaginer le fonctionnement simultané de 1 024 comparateurs à fréquence élevée. Les comparateurs les plus sensibles, correspondant aux bits les moins significatifs, doivent être insensibles aux actions se déroulant pour les bits de poids forts. Les parasites de commutation doivent être sans effet sur les comparateurs les plus sensibles. La précision du codage est en fait inférieure à celle donnée par la théorie de l'erreur de quantification exposée précédemment. Les constructeurs définissent alors un nombre de **bits effectifs** qui est simplement le nombre de bits de codage qu'il faudrait pour obtenir le rapport signal sur bruit de quantification réellement atteint. Ce nombre est toujours plus faible que le nombre de bits de codage. Pour le calculer, il suffit de reprendre la formule :

$$\zeta^2 eff = 1{,}77 + 6{,}02 n_{eff}$$

$n_{eff}$ est le nombre de bits effectifs et $\zeta$ est le rapport signal sur bruit de quantification. Ce nombre est donné pour une sinusoïde en entrée, de fréquence déterminée.

## Convertisseur à approximations successives

Présentant un temps de conversion plus élevé que le convertisseur parallèle, le convertisseur à approximations successives offre l'avantage d'un plus grand nombre de bits de conversion possible et d'une plus faible consommation. Il est donc utilisé dans toutes les applications ne demandant pas une fréquence d'échantillonnage élevée (inférieure à 100 kHz). Le principe de fonctionnement est représenté figure 7.10, il s'apparente à la méthode de la pesée à l'aide d'une balance à plateaux. La tension d'entrée est comparée à des tensions de référence s'approchant de plus en plus de la valeur à coder. À chaque pesée, la tension d'entrée est comparée à une tension de référence. Lorsqu'il y a dépassement, un poids plus faible est appliqué à la pesée suivante. Quand il n'y a pas dépassement, le poids est maintenu et un poids supplémentaire plus faible est ajouté pour la pesée suivante. Les tensions de référence (ou poids) sont successivement : $V_{REF}/2$, $V_{REF}/4$, $V_{REF}/8$, etc.

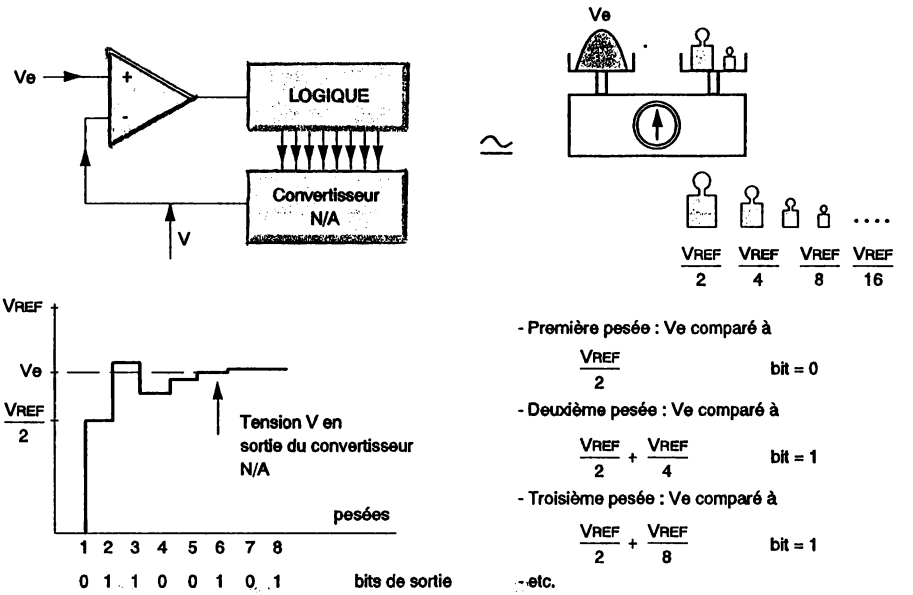

Ve
LOGIQUE
Convertisseur N/A
V
Ve
VREF/2 VREF/4 VREF/8 VREF/16
VREF
Ve
VREF/2
Tension V en sortie du convertisseur N/A
pesées
1 2 3 4 5 6 7 8
0 1 1 0 0 1 0 1
bits de sortie
- Première pesée : Ve comparé à VREF/2 bit = 0
- Deuxième pesée : Ve comparé à VREF/2 + VREF/4 bit = 1
- Troisième pesée : Ve comparé à VREF/2 + VREF/8 bit = 1
- etc.

Fig. 7.10
Convertisseur à approximations successives.

Au fur et à mesure que les pesées s'effectuent, la valeur binaire obtenue s'approche du code binaire correspondant à la tension d'entrée. Dans l'exemple donné (conversion sur huit bits), au bout de huit pesées (huit comparaisons), la valeur binaire est obtenue sur huit bits. Les temps de conversion sont de l'ordre de dix microsecondes pour douze bits de codage et de deux microsecondes pour huit bits. Ces convertisseurs réalisent donc un excellent compromis entre la précision de quantification et la rapidité du codage.

La figure 7.11 illustre le déroulement temporel des opérations et montre la formation séquentielle des différents bits de conversion. De la même manière que pour le convertisseur parallèle, il est nécessaire de commander le convertisseur par un signal logique pour démarrer le processus. Une horloge interne est alors mise en marche et séquence les pesées successives. Un état logique est fourni par le convertisseur pour indiquer la fin de la conversion et la validité des bits de codage.

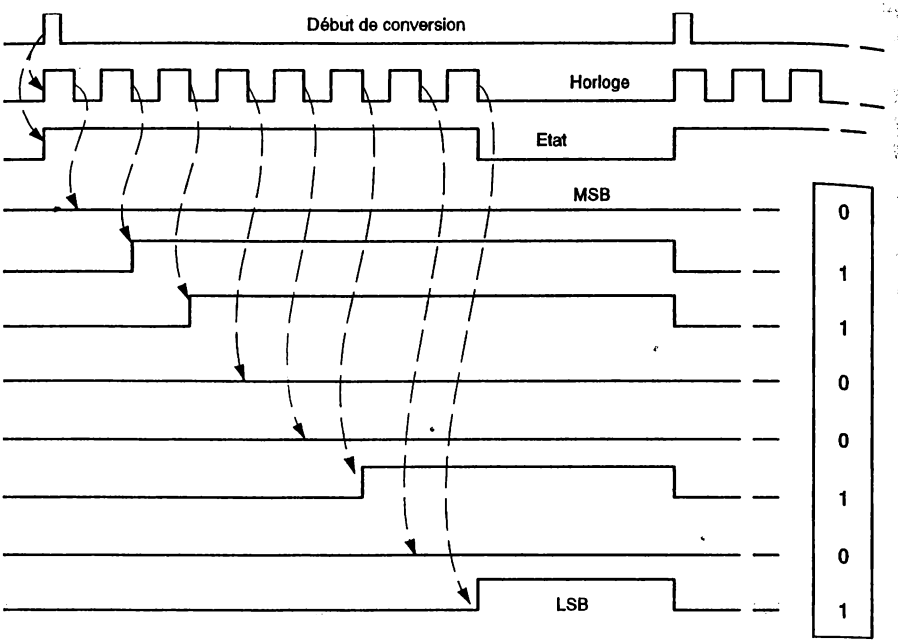


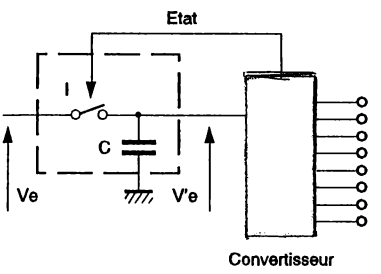

Fig. 7.11
Convertisseur à approximations successives, chronogramme.

Les bits de codage sont disponibles sous forme série ou sous forme parallèle puisqu'ils sont créés successivement au rythme de l'horloge interne. Le fonctionnement de ce convertisseur permet d'introduire une nouvelle fonction : l'**échantillonneur-bloqueur**. En effet, les diverses comparaisons ne sont pas effectuées en même temps. Que se passe-t-il alors si le signal d'entrée varie ? Dans ce cas, si aucune précaution n'est prise, la valeur codée est une sorte de mélange entre les diverses valeurs possibles du signal pendant le temps de conversion. Le but recherché n'est donc pas atteint : obtenir la valeur codée du signal à un instant donné. Pour remédier à ce défaut, il est nécessaire d'ajouter en entrée de convertisseur un circuit spécial appelé échantillonneur-bloqueur. Son rôle est de conserver la valeur analogique du signal d'entrée pendant tout le temps de conversion. Ce circuit est schématisé figure 7.12.

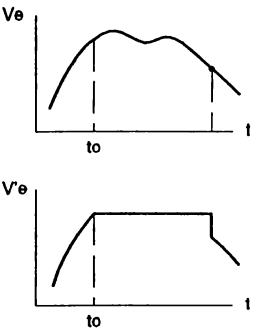
Pendant la conversion, l'interrupteur *I* est ouvert et le signal d'entrée est mis en mémoire aux bornes de la capacité *C*. Quand la conversion est achevée, l'interrupteur est à nouveau fermé et le signal est appliqué au convertisseur. L'interrupteur *I* est piloté par l'état de sortie du convertisseur. L'incertitude sur l'instant de la

conversion (**jitter**) introduit un terme d'erreur quand le signal varie. Si la tension d'entrée varie avec une pente $\frac{dV}{dt}$, une incertitude $\Delta t$ du temps d'échantillonnage induit une incertitude $\Delta V$ sur la mesure de la tension :

$$\Delta V = \left(\frac{dV}{dt}\right)_{t_0} \Delta t$$




7.12.a




7.12.b

Fig. 7.12
Échantillonnage bloqueur associé à un convertisseur.

Prenons un exemple : un signal sinusoïdal de 1 V de fréquence 100 kHz est échantillonné sur douze bits. Quelle doit-être la précision sur le temps d'échantillonnage pour conserver la précision des douze bits ?

La pente du signal est $2\pi f \cdot V_e \cos(2\pi ft)$, elle est maximale en $t = 0$. Sa valeur numérique est alors pour $V_e = 1$ V et $f = 100\,000$ Hz :

$$\left(\frac{dV}{dt}\right)\max = 6{,}28 \times 10^5 \; V \times s^{-1}$$

Si l'on désire conserver la précision des douze bits, on suppose alors que le signal est codé à pleine échelle soit au canal 4096 pour l'amplitude maximale de 1 V. L'erreur de quantification mesurée par son écart-type est alors :

$$\sigma_v = \frac{1}{4096} \times \frac{1}{\sqrt{12}}$$

On en déduit donc l'écart type correspondant sur le temps :

$$\sigma_t = \frac{1}{4096} \times \frac{1}{\sqrt{12}} \times \frac{1}{6{,}28 \times 10^5} \approx 100 \text{ ps}$$

La précision sur l'instant d'échantillonnage doit donc être excellente.

## Convertisseur à rampe

Quand une excellente précision est demandée, par exemple pour faire des appareils de mesure, un autre principe de mesure est mis en application : la conversion à rampe. Le principe est indiqué figure 7.13.

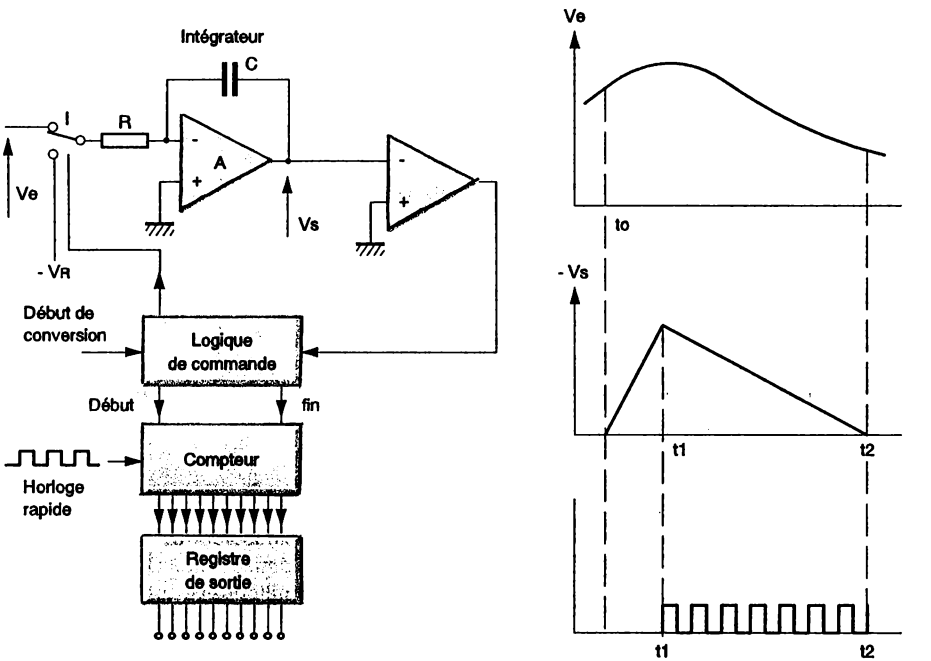



Fig. 7.13
Convertisseur à rampe.

La tension d'entrée charge un condensateur qui est ensuite déchargé avec une constante de temps assez longue. La mesure du temps de décharge $t_2 - t_1$ à l'aide d'un compteur est alors proportionnelle à la tension d'entrée. La sortie du compteur donne ainsi directement la valeur binaire de codage. L'étude détaillée du fonctionnement de ce convertisseur n'est pas indispensable lors d'une première lecture. Elle constitue cependant un bon exercice d'application des méthodes de calcul des circuits.

Le signal d'entrée est appliqué sur un étage intégrateur formé classiquement par un amplificateur opérationnel contre-réactionné par une capacité $C$ (voir chapitre 11 tome 1). Pendant la première phase de $t_0$ à $t_1$, le signal est intégré et crée une tension de sortie $V_s(t)$ :

$$-V_s(t)=\frac{1}{RC}\int_0^t V_e(t')dt'$$

Sa valeur en $t_1$ est alors :

$$-V_s(t_1)=\frac{V_e}{RC}(t_1-t_0)$$

À partir de $t_1$, une deuxième phase se déclenche et l'interrupteur *I* est positionné pour appliquer en entrée une référence $-V_R$. En considérant l'amplificateur comme idéal, on obtient :

$$\frac{V_R}{R}=CpV_S$$

soit

$$V_S(t)=\frac{-1}{RC}\int_{t_1}^t V_R dt+V_S(t_1)$$

La capacité se décharge donc avec une constante de temps égale à *RC*. La tension $V_S$ atteint la valeur 0 au temps $t_2$ donné par la relation :

$$0=\frac{(t_2-t_1)}{RC}V_R-\frac{(t_1-t_0)}{RC}V_e$$

soit :

$$t_2-t_1=\frac{V_e}{V_R}(t_1-t_0)$$

La différence de temps $t_2-t_1$ est donc proportionnelle à la tension d'entrée. Il suffit donc de coder cet intervalle de temps, par exemple en déclenchant une horloge sur $t_1$ et en l'arrêtant sur $t_2$, pour obtenir la valeur codée de l'intervalle de temps. Le comptage des impulsions de l'horloge réalise ce codage. La précision du codage est d'autant meilleure que la fréquence de l'horloge est élevée et que le temps de la mesure $t_2-t_1$ est élevé.

Cet exemple montre une fois encore le compromis entre la précision et le temps de mesure. Un échantillonneur-bloqueur en entrée du convertisseur permet de fixer avec précision l'instant d'échantillonnage. Ce type de convertisseur permet d'obtenir des codages de grande qualité, en particulier si on considère la linéarité de la mesure. Le temps de codage est long, généralement de l'ordre de la milliseconde.

## Convertisseur sigma-delta

Apparu plus récemment, le convertisseur sigma-delta permet d'obtenir des précisions de codage que ne permettent pas les autres types de convertisseur. Le principe de conversion est assez complexe et tire parti des techniques numériques de traitement du signal.

Le signal d'entrée est échantillonné à une fréquence très supérieure à celle qui serait nécessaire par application de la règle de Shannon.

Le signal est ensuite filtré afin de minimiser le bruit de quantification. La linéarité de la conversion est également excellente. Des précisions aussi élevées que 24 bits sur une dynamique de 100 dB sont possibles. Ces convertisseurs sont réalisés sous forme de circuits intégrés monolithiques et sont utilisés en particulier dans l'enregistrement numérique sur disque laser.

## Récapitulatif des convertisseurs analogique-numérique

Après l'exposé des diverses techniques de conversion, une certaine impression de confusion peut naître dans l'esprit du lecteur. La question à laquelle il doit répondre est : quel type de convertisseur dois-je utiliser dans mon application ?

Les deux facteurs principaux pris en compte dans la classification proposée sont le nombre de bits de codage et la fréquence d'échantillonnage, choisis selon les critères exposés précédemment. D'autres critères entrent également en jeu : la consommation et le coût du circuit.

| | |
|---|---|
| **Précision**<br>**Fréquence d'échantillonnage**<br>**Consommation**<br>**Coût** | ⇒ **CHOIX DU CONVERTISSEUR** |

La figure 7.14 résume les propriétés des diverses familles offertes par l'industrie électronique en fonction des deux premiers critères : le nombre de bits et la fréquence d'échantillonnage. En abscisse est portée la fréquence d'échantillonnage maximale, et en ordonnée le nombre maximum de bits de codage.

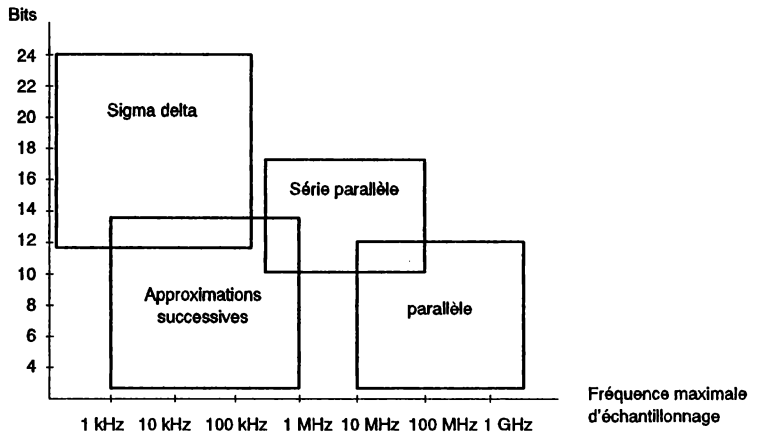



Fig. 7.14
Choix du type de convertisseur.

Les convertisseurs série-parallèle complètent ce tableau. Ils fonctionnent selon une combinaison variable des techniques exposées précédemment.

La figure 7.15 représente un convertisseur associant une technique parallèle sur un nombre de bits réduit (huit dans le cas de la figure) à une technique série. Dans ce cas, un signal de différence est formé entre le signal à coder et la valeur correspondant à la conversion grossière effectuée sur un nombre limité de bits. Ce signal de différence est, dans une deuxième passe, amplifié et codé avec le même convertisseur parallèle. Deux conversions consécutives sur huit bits permettent alors une conversion totale sur seize bits.

Les convertisseurs exposés précédemment sont le plus souvent monolithiques, c'est-à-dire réalisés sur une seule puce de silicium. Le coût est alors modéré. Quand le convertisseur est réalisé par un assemblage de puces et de composants discrets, généralement sous forme de circuits hybrides sur un support céramique, le coût est alors plus élevé. Ce choix ne peut se justifier que s'il apporte une réelle augmentation de performances.

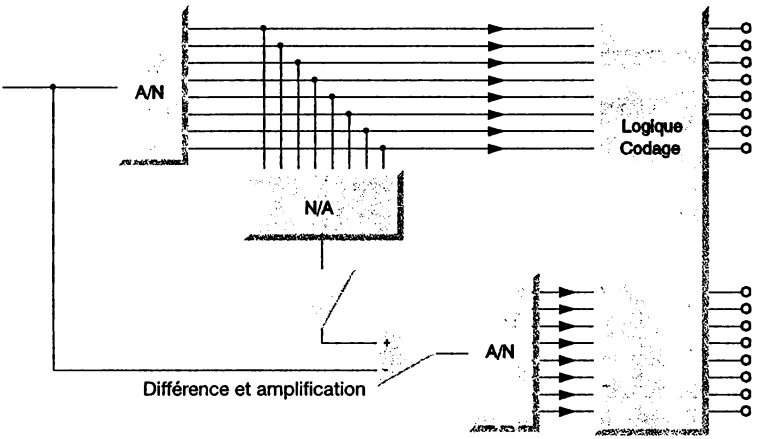



Fig. 7.15
Convertisseur mixte.

## Convertisseur numérique-analogique

Dans un certain nombre d'architectures de convertisseurs, nous avons rencontré la fonction inverse : la conversion numérique-analogique. Elle permet de générer, à partir d'un mot binaire, le signal analogique correspondant. La conversion mixte est un exemple d'utilisation de cette fonction. Dans de nombreuses autres applications, la conversion numérique-analogique est nécessaire. Pensons, par exemple, à la lecture des disques laser, à la synthèse vocale, à la création d'une image sur un écran permettant de visualiser les résultats d'un logiciel de calcul, au fonctionnement d'un oscilloscope numérique.

Plus simple à réaliser que la conversion analogique-numérique, cette fonction mérite cependant quelques explications. La fonction de transfert d'un convertisseur numérique-analogique est l'inverse de celle d'un convertisseur analogique-numérique mais à la différence de ce dernier, elle n'est pas affectée d'une erreur de quantification. La valeur affectée à un code binaire donné peut en effet, en théorie du moins, être la valeur exacte correspondant à ce code. L'erreur de quantification est créée au moment de la génération de la valeur codée, qu'elle soit un résultat de calcul ou le résultat d'une conversion analogique-numérique. Il suffit alors de garantir une bonne linéarité dans la conversion et une absence de signaux parasites dûs aux commutations (*glitches*) pour assurer un bon fonctionnement du convertisseur. La figure 7.16 représente la fonction de transfert idéale d'un convertisseur numérique-analogique trois bits. Des parasites peuvent apparaître sur le

signal de sortie. Ils sont en général créés par les variations brusques de courant quand le code varie. Pour éliminer la forme en marche d'escalier inhérente à la quantification, un lissage du signal est opéré. Ce lissage est par exemple le remplacement d'une marche d'escalier par un segment de droite. Les parasites de commutation ne sont pas toujours totalement éliminés par le lissage.

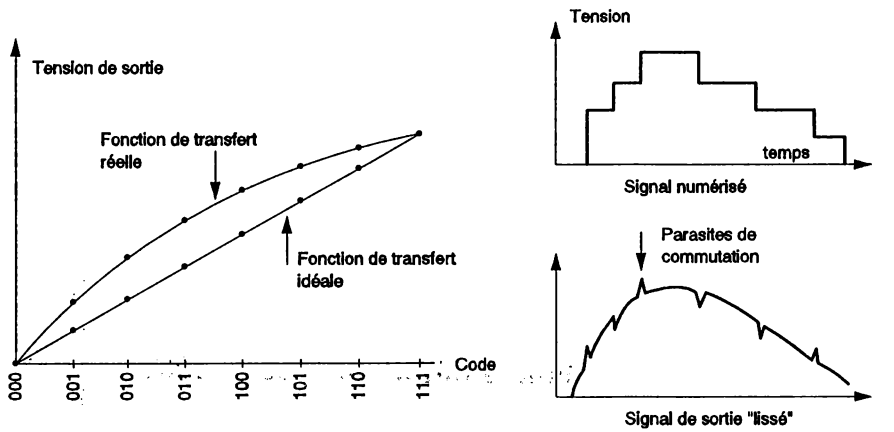
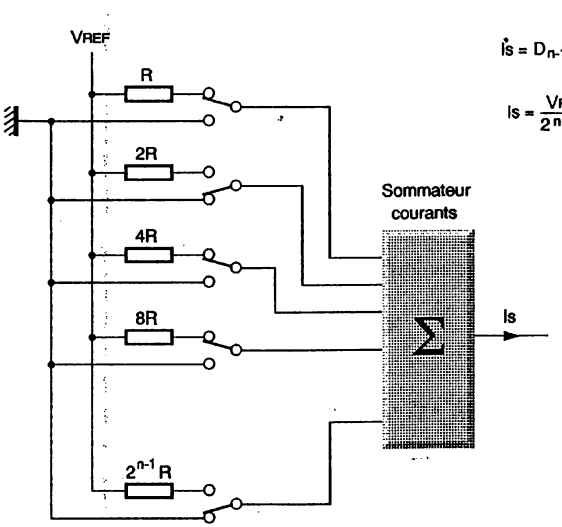



Fig. 7.16
Fonction de transfert d'un convertisseur numérique-analogique.




$$Is = D_{n-1} \cdot \frac{V_{REF}}{R} + \ldots + D1 \cdot \frac{V_{REF}}{2^{n-2} R} + D0 \cdot \frac{V_{REF}}{2^{n-1} R}$$

$$Is = \frac{V_{REF}}{2^{n-1} R} (D0 + D1 \cdot 2^1 + D2 \cdot 2^2 + \ldots + D_{n-1} \cdot 2^{n-1})$$

avec Di = 0 ou Di = 1

Fig. 7.17
Convertisseur analogique-numérique.

Le principe général de fonctionnement d'un convertisseur numérique-analogique est représenté figure 7.17. La méthode consiste à commuter des courants en fonction du code binaire. Un sommateur réalise la conversion en additionnant les divers courants correspondant aux différents bits. Un amplificateur opérationnel peut réaliser la fonction, comme il a été expliqué au chapitre 11 du tome 1. Dans la pratique, le principe précédent est réalisé assez fréquemment par un montage en échelle, également appelé réseau *R/2R*. Ce circuit est représenté figure 7.18.

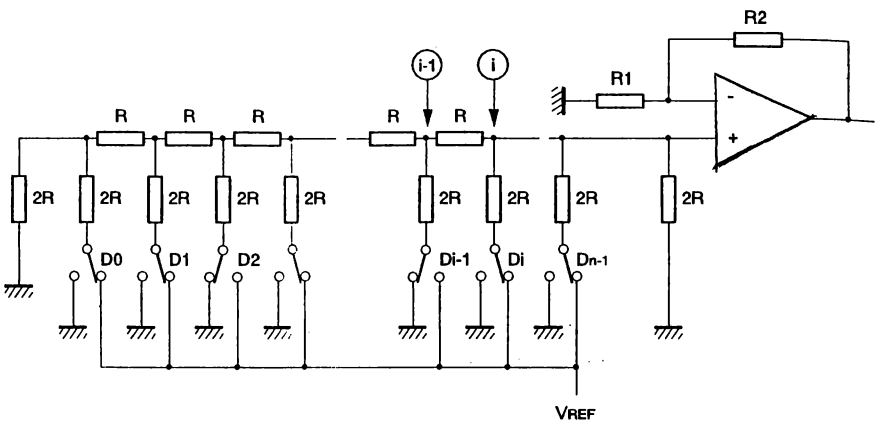



Fig. 7.18
Convertisseur analogique-numérique en échelle.

Pour comprendre le fonctionnement de ce montage, il faut d'abord remarquer qu'en tout nœud du réseau, l'impédance vue est toujours $2R$ que l'on regarde à droite, à gauche ou vers le commutateur. La propriété peut se prouver par récurrence sur le nombre de cellules. On suppose que la propriété est vraie au nœud $i$, il est ensuite facile de prouver qu'elle est vraie au nœud $i + 1$. Dans une deuxième étape, on peut également prouver par récurrence que la tension en sortie s'écrit bien :

$$V_S = \frac{R_1 + R_2}{R_1} \times \frac{V_{REF}}{2^{n-1}} (D_0 + 2D_1 + 2^2 D_2 + ..... + 2^{n-1} D_{n-1})$$

La démonstration rigoureuse de cette formule est laissée au lecteur amateur de démonstrations mathématiques ; elle n'offre pas un grand intérêt d'un point de vue strictement électronique. Des convertisseurs douze bits sont disponibles avec un temps d'établissement de 30 ns. Il est donc possible de reconstituer un signal variant à 15 MHz. Le temps d'établissement est le temps minimum séparant l'application des bits en entrée de la formation d'un signal analogique stable en sortie correspondant à cette

valeur binaire. Des convertisseurs plus rapides sont également fabriqués et des temps d'établissement de quelques nanosecondes sont possibles avec un nombre de bits plus limité. Ces convertisseurs fournissent une valeur du signal au fur et à mesure que le code binaire d'entrée est appliqué. Ils réalisent donc une conversion numérique-analogique en temps réel, c'est-à-dire au fur et à mesure que les valeurs binaires apparaissent.

## Interfaces avec les microprocesseurs

Il ne suffit pas d'obtenir convenablement la valeur binaire correspondant à une valeur analogique, encore faut-il pouvoir l'utiliser. Deux cas se rencontrent en pratique :

– le premier, le plus fréquent, consiste à présenter les bits correspondant à chaque échantillon en entrée d'un microprocesseur, d'un microcontrôleur ou d'un processeur de signal (**DSP**).
En pratique, il suffit de se connecter sur le bus du processeur (voir le chapitre 4 sur les microprocesseurs et microcontrôleurs). La contrainte principale apparaît alors immédiatement : le temps de calcul doit être suffisamment rapide pour que le processeur puisse accepter les données au fur et à mesure qu'elles sont produites par le convertisseur ;
– le second cas est généralement réservé aux fréquences d'échantillonnage élevées.
Il consiste à écrire les données dans une mémoire rapide au fur et à mesure qu'elles se présentent. Le microprocesseur a alors tout son temps pour analyser et traiter les données en mémoire dans une deuxième phase. Pendant ce temps, il ne sera plus possible d'accepter d'autres données et seule une faible fraction du temps sera analysée. Prenons par exemple le cas d'un oscilloscope numérique : le signal à visualiser est échantillonné pendant un temps relativement court correspondant à la tranche de temps à visualiser. Après mise en mémoire des échantillons, une deuxième phase permet de convertir ces valeurs enregistrées en tensions successivement appliquées aux plaques de déviation du tube cathodique pour positionner le spot.

La figure 7.19 résume graphiquement ces deux modes de fonctionnement.

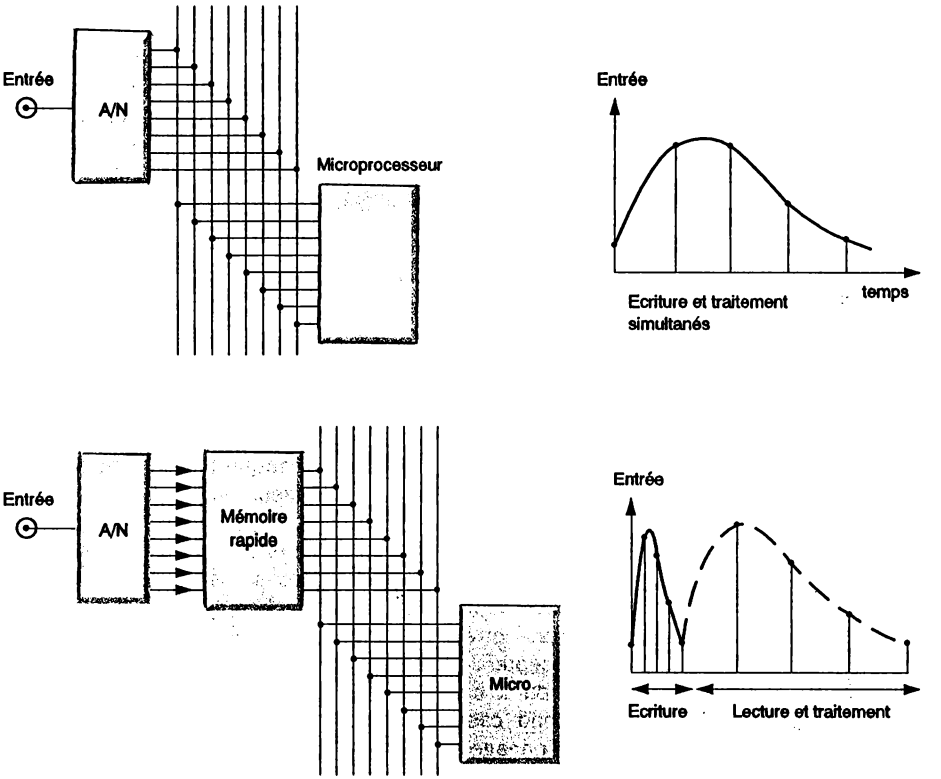



Fig. 7.19
Utilisation d'un convertisseur analogique-numérique.

Il est maintenant possible de voir de manière détaillée le fonctionnement des échanges entre un convertisseur et un microprocesseur. Ce paragraphe fera souvent référence au chapitre 4 cité plus haut. Les opérations sont très détaillées car la compréhension des échanges de données est un des éléments les plus importants dans la conception des systèmes. La figure 7.20 représente les circuits intervenant dans cet échange et les signaux émis.

On y trouve le microprocesseur avec ses **lignes de données** (huit lignes dans l'exemple soit huit bits), ses **lignes d'adresses** (seize lignes dans l'exemple traité), et deux signaux de contrôle émis par le microprocesseur (*WR* et *RD*) dont le rôle sera précisé par la suite.

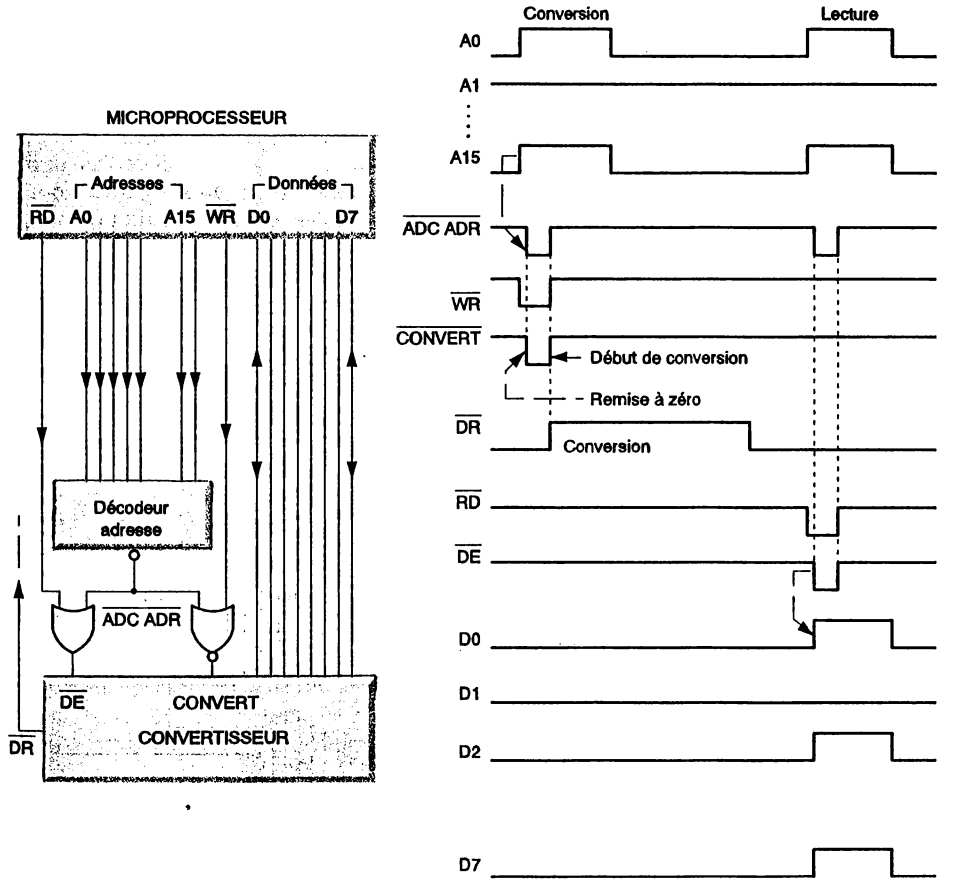



Fig. 7.20
Interface entre un convertisseur et un microprocesseur.

Le convertisseur communique avec l'extérieur par huit lignes de données également. Deux entrées permettent de démarrer la conversion (*CONVERT*) et de placer les bits de conversion sur les lignes de sortie (*DE*, *data enable*). Un signal émis par le convertisseur (*DR*, *data ready*) indique que la conversion est terminée et que les bits de la valeur codée sont disponibles. Un circuit de décodage d'adresse fonctionne selon les principes exposés au chapitre 1 et permet au convertisseur de reconnaître qu'il est concerné par la transaction. Quelques portes OU complètent le schéma. Par convention, la notation $\overline{X}$ sur un schéma signifie que la fonction est réalisée sur l'état logique bas.

Voyons maintenant les différentes étapes dans l'échantillonnage d'un signal, la conversion de l'échantillon en binaire et la présentation des bits de conversion en entrée du microprocesseur.

## 1 Étape 1 : Échantillonnage du signal

Le microprocesseur doit envoyer un signal au convertisseur afin que celui-ci échantillonne le signal et démarre la conversion. Le programme en mémoire initialise les opérations. Tout d'abord, il commande au microprocesseur de placer sur les lignes d'adresse une valeur qui identifie le convertisseur (par exemple 1010110001). Un signal $\overline{WR}$ est émis simultanément, il est actif à l'état bas dans l'exemple donné et indique que c'est une opération d'écriture qui est en cours. Le convertisseur doit reconnaître qu'il est concerné par l'opération et doit donc « décoder » l'adresse en la comparant à son adresse inscrite en « dur ». Dans la pratique, une simple comparaison binaire est effectuée selon le schéma indiqué figure 7.21.

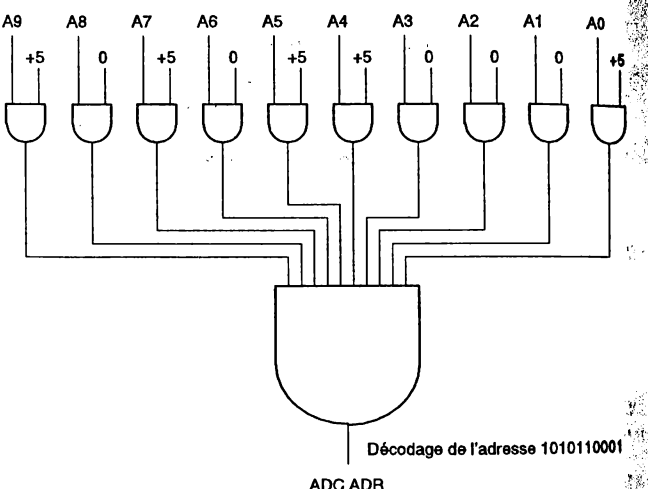



Fig. 7.21
Décodage d'adresse.

La valeur inscrite en dur est simplement définie par la position des interrupteurs fixant les niveaux en entrée des portes de comparaison. Un état $\overline{ADCADR}$ est élaboré par le décodeur quand les deux mots binaires coïncident. Cet état est mis en « ET » logique avec le signal d'écriture $\overline{WR}$ pour créer le signal $\overline{CONVERT}$ permettant de démarrer la conversion. Il faut noter que ce cycle est du point de vue du microprocesseur un cycle d'écriture puisque le processeur écrit dans le périphérique concerné, en

l'occurrence le convertisseur, l'ordre de démarrer une opération. Il faut également remarquer que la fonction « ET » est en fait réalisée par un « OU » logique. Il suffit d'écrire l'équation logique suivante pour s'en convaincre :

$$\overline{CONVERT} = \overline{WR} + \overline{ADCADR}$$

soit :

$$CONVERT = WR \cdot ADCADR$$

## 2 Étape 2 : Démarrage de la conversion

Le front montant de l'impulsion *CONVERT* est utilisé pour remettre à zéro les circuits logiques nécessaires au bon fonctionnement du convertisseur. Le front descendant de *CONVERT* démarre l'échantillonnage du signal et la conversion. Quand elle est terminée, le convertisseur élabore un état $\overline{DR}$ actif à l'état bas.

## 3 Étape 3 : Lecture de la valeur de l'échantillon par le processeur

Deux cas sont alors possibles : ou bien le temps de conversion est suffisamment connu et le microprocesseur se contente d'envoyer au convertisseur, après un retard suffisant, l'ordre de placer les valeurs sur ses lignes de données ou bien le microprocesseur n'est pas bien assuré du temps de conversion et il utilise alors l'information de l'état $\overline{DR}$ pour démarrer la phase de lecture. L'état $\overline{DR}$ pourra être lu par le convertisseur ou bien être envoyé sur une ligne d'interruption. Nous ne détaillerons pas ces deux manières de procéder qui dépendent plus du programme lui-même et du type de processeur que des considérations générales données dans ce chapitre. Quand le processeur juge que la lecture est possible, il envoie un ordre de lecture au convertisseur ($\overline{RD}$). De la même manière que dans la phase d'écriture de l'ordre de conversion, cet ordre $\overline{RD}$ est mis en ET logique avec l'adresse du convertisseur pour indiquer que c'est bien lui qui est concerné par la lecture. Une impulsion $\overline{DE}$ est alors élaborée. Elle autorise le convertisseur à placer les états logiques correspondant au codage de l'échantillon sur les lignes de données bidirectionnelles du microprocesseur. Les circuits de sortie du convertisseur doivent être des circuits dits à trois états (voir le chapitre 3 de cet ouvrage). Quand le convertisseur ne reçoit pas l'ordre de placer des états logiques sur les lignes de données, il doit laisser ces lignes disponibles pour d'autres opérations. D'un point de vue électrique, il laisse ses sorties dans un état à haute impédance de telle sorte que la charge électrique qu'elles représentent soit négligeable.

Cette description a pu paraître un peu fastidieuse. Elle est d'un usage si courant dans l'utilisation des microprocesseurs et des convertisseurs qu'elle méritait bien ces explications détaillées.

## Caractéristiques des convertisseurs et critères de choix

Dans ce paragraphe, nous allons passer en revue les caractéristiques principales des convertisseurs. Ces caractéristiques sont données dans les « **Data books** » des constructeurs et permettent à l'utilisateur de faire son choix. Quelques produits seront donnés en exemple.

### 1 Précision (accuracy)

C'est simplement le nombre de bits fournis par un CAN (analogique-numérique) ou acceptés en entrée par un CNA (numérique-analogique). La notion de bits effectifs est introduite pour chiffrer la dégradation de performances du convertisseur quand la fréquence du signal analogique d'entrée augmente.

### 2 Nombre de conversions par seconde ou fréquence d'échantillonnage (sampling frequency)

C'est, pour un CAN, le nombre maximum de conversions par seconde. Le signal est donc échantillonné à cette fréquence. La règle de Shannon impose alors au signal d'entrée d'avoir une fréquence maximale inférieure à la moitié de cette fréquence. Cette fréquence limite est appelée **fréquence de Nyquist**. Pour un CNA, on définit la fréquence maximale des variations des bits en entrée.

### 3 Rapport signal sur bruit et bits effectifs (signal to noise ratio and effective bits)

Exprime le rapport entre la valeur efficace du signal et l'écart type du bruit de quantification. Quand la fréquence du signal analogique d'entrée augmente, ce rapport se dégrade et tout se passe comme si le convertisseur fournissait un nombre de bits plus faible.

### 4 Gamme du signal d'entrée (range)

Plage de tension à l'intérieur de laquelle le signal d'entrée doit être compris. Elle s'étend généralement de 0 à 10 V ou de – 5 à 5 V.

### 5 Précision d'échantillonnage (aperture jitter)

Mesure l'incertitude sur l'instant d'échantillonnage. Elle est quantifiée par l'écart type de la distribution. Cette caractéristique n'est donnée que si l'échantillonneur bloqueur est inclus dans le convertisseur.

## 6 Non linéarité intégrale et non linéarité différentielle (integral non linearity, differential non linearity)

Ces caractéristiques mesurent les imperfections de la fonction de transfert et sont représentées figure 7.22.

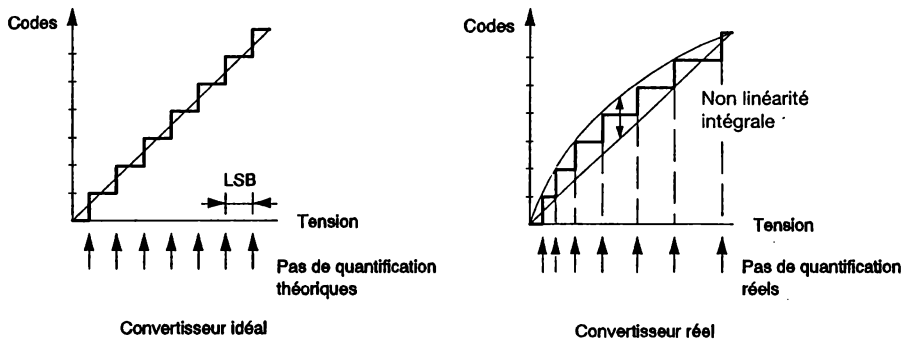



Fig. 7.22
Non-linéarité intégrale et non-linéarité différentielle.

La non linéarité intégrale est l'écart maximum entre la valeur correspondant à la valeur codée et la valeur théorique donnée par la droite idéale représentant la fonction de transfert. Elle s'exprime en nombre de LSB. Le LSB ou quantum élémentaire est le rapport entre la tension maximale en entrée et le nombre de pas de quantification, soit $2^n$ où $n$ est le nombre de bits.

La non linéarité différentielle est l'écart maximum entre la valeur correspondant à un pas de quantification et sa valeur théorique. Elle s'exprime également en LSB. Dans l'exemple donné, la valeur maximale est 2 LSB.

## 7 Consommation (power consumption)

C'est la puissance consommée par le convertisseur.

## Quelques exemples de produits

| | AD1776 | AD9012 | AD7582 | DSP56ADC16 |
|---|---|---|---|---|
| Précision (bits) | 14 | 8 | 12 | 16 |
| Conversions/s | 100 kHz | 100 MHz | 10 kHz | 100 kHz |
| Bits effectifs | | 7,5 à 12 MHz | | |
| Gamme entrée | 0 à 10 V | 0 à 10 V | 0 à 5 V | 0 à 5 V |
| Précision échantillonnage | 150 ps | 15 ps | | |
| Non linéarité intégrale | ±1 LSB | ±0,6 LSB | ±1 LSB | |
| Non linéarité différentielle | | ±0,6 LSB | ±1 LSB | |
| Consommation | 560 mW | 950 mW | 75 mW | |
| Type | mixte série-parallèle | parallèle | approximations successives | sigma-delta |

ENTRAÎNEMENT

**1** Un signal varie entre 10 kHz et 1 MHz. Choisir la fréquence d'échantillonnage pour conserver l'information.

**2** Une sinusoïde de fréquence 50 Hz est échantillonnée à 100 Hz pour respecter le théorème de Shannon. Les instants d'échantillonnage sont 0, 100 ms, 200 ms, etc. Les échantillons sont tous nuls. Comment reconstituer le signal ?

**3** Un signal varie entre 1 mV et 1 V. On veut le coder avec une précision de 1 % (écart type). Combien de bits de codage sont-ils nécessaires ?

Si on applique avant le codage une fonction de transfert non linéaire du type :

$$\frac{V_S}{V_E} = K \frac{LOG\left(1 + \frac{\mu V_E}{V_{max}}\right)}{(1+\mu)}$$

dans cette formule, $K$, $V_{max}$ et $\mu$ sont des paramètres. Montrez qu'il est alors possible de réduire le nombre de bits de codage en gardant la précision sur toute la gamme. Choisissez les paramètres.

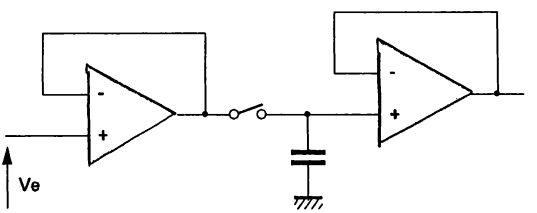
**4** Imaginez un voltmètre numérique mesurant entre 1 mV et 1 000 V sur sept gammes de mesure.

Comment choisir le convertisseur pour conserver une précision de $10^{-3}$ ?

Donnez le schéma du voltmètre.

**5** Comment réaliser un codeur parallèle de 7 bits avec deux codeurs de 6 bits ?

**6** Montrez que le schéma suivant permet de réaliser un meilleur échantillonneur bloqueur que celui de la figure 7.12.




**7** Comparez la technique série-parallèle à celle de l'exercice 5.

**8** Retrouvez la formule du convertisseur $R/2R$.

**9** Représentez le schéma de principe d'un oscilloscope numérique. Donnez les caractéristique des convertisseurs AN et NA utilisés.

**10** Le convertisseur donne un résultat sur 12 bits. Le bus de lecture est sur 8 bits. Comment lire le convertisseur ? Donnez le schéma de la logique.

Chapitre 8

# SYNTHÉTISEUR DE FRÉQUENCE

## SOMMAIRE


- Analyse qualitative du fonctionnement de la boucle
- Le filtre de boucle
- Plage de capture et plage de verrouillage
- Modulation d'une boucle à verrouillage de phase
- Démodulation par une boucle à verrouillage de phase
- Bruit de phase
- Éléments constituant une boucle à verrouillage de phase


*Le **synthétiseur de fréquence** est employé principalement lorsqu'on veut disposer d'une fréquence stable et parfaitement connue. Cette structure que l'on retrouve sous le nom de boucle à accrochage ou verrouillage de phase, ou encore PLL pour Phase Locked Loop, est une des plus importantes de l'électronique, et ses applications couvrent un vaste domaine.*

*Dans les appareils de transmission professionnels et grand public, les PLL sont utilisés pour :*

*– la génération des porteuses en émission et la génération des oscillateurs locaux en réception,*

*– la démodulation des signaux analogiques modulés en fréquence,*

*– la démodulation des signaux logiques en FSK,*

*– les systèmes de récupération d'horloge en transmission numérique.*

*En métrologie, les PLL sont utilisés pour générer des signaux de fréquence parfaitement connus et stables.*

*Imaginons un cas concret dans lequel on veuille disposer d'un signal à la fréquence de 1 000 MHz, précise à 1/10 000. Pour générer ce signal, on a donc recours à un oscillateur. À cette fréquence, l'oscillateur ne peut être que du type LC. Un oscillateur LC a un circuit oscillant constitué par une self et un condensateur mis soit en série soit en parallèle. Les valeurs de selfs et capacités réalisables physiquement limitent le coefficient de surtension du circuit oscillant.*

*Pour un circuit RLC série, le coefficient de surtension Q du circuit vaut :*

$$Q = \frac{L\omega}{R}$$

*Pour un circuit RLC parallèle, le coefficient de surtension Q du circuit vaut :*

$$Q = RC\omega$$

*Dans ces deux cas, ω vaut :*

$$\omega = \frac{1}{\sqrt{LC}}$$

*Il n'est pas possible de réaliser directement un oscillateur à quartz en mode fondamental en général limité à 30 MHz.*
*Un quartz est un composant qui peut être représenté par un circuit RLC série shunté par une seconde capacité C'. Ce schéma R, L, C, C', n'est qu'un schéma* ***équivalent.***
*Par construction, la valeur de L est très élevée et la valeur de C est très faible, ce qui procure un cœfficient de surtension élevé. Ce coefficient confère à l'oscillateur une grande stabilité.*
*Pour des fréquences supérieures, il existe des résonateurs à onde acoustique de surface, dont le schéma équivalent est identique à celui du quartz, mais dont le calage en fréquence ne peut être aussi précis que celui du quartz. L'emploi de l'oscillateur LC est alors inévitable.*
*Les dérives en fréquence d'un oscillateur LC sont fonction notamment du temps et de la température agissant tant sur les éléments actifs que sur les éléments passifs. Le calage en fréquence est fonction de ces mêmes éléments. Le problème réside dans la stabilisation de cet oscillateur. la boucle à verrouillage de phase résout le problème. L'oscillateur est alors remplacé par un* ***oscillateur contrôlé en tension VCO****, les éléments annexes n'ont pour seul objet que la stabilisation en fréquence du VCO.*

## Analyse qualitative du fonctionnement de la boucle

La boucle à verrouillage de phase est un système asservi. Dans tout asservissement il s'agit, à un instant donné, de comparer une grandeur à la grandeur de référence. De cette comparaison il résulte une indication destinée à la correction de la grandeur comparée. Tant que les deux grandeurs ne sont pas égales, une correction est envoyée. Dès que les grandeurs sont égales, aucune correction n'est plus nécessaire jusqu'à ce que les deux grandeurs diffèrent à nouveau.

Dans le cas d'une application à la boucle à verrouillage de phase, il s'agit d'asservir un oscillateur à une fréquence de référence. À l'instant de la mesure, le comparateur de phase peut donner trois types d'information :
- la phase à asservir est en avance sur la référence,
- la phase à asservir est en retard sur la référence,
- les deux signaux sont en phase. Dans ce cas, les deux fréquences sont identiques.

Dans le cas d'un comparateur de phase numérique, ces trois cas se concrétisent de la manière suivante :
- avance : impulsion positive,
- retard : impulsion négative,
- égalité : pas d'impulsion.

L'oscillateur à asservir est commandé par une tension continue. Un filtre passe-bas est intercalé entre la sortie du comparateur de phase et l'entrée de commande de l'oscillateur. Ce filtre a un effet de lissage. Il moyenne les impulsions de sortie du comparateur et les transforme en une tension continue apte à la commande de l'oscillateur. Tant que les deux fréquences ne sont pas en phase, le comparateur de phase associé au filtre génère une tension d'erreur.

## 1 Principe

Comme dans le chapitre consacré aux procédés de modulation, on considère non pas des signaux sinusoïdaux purs mais des signaux représentés par des fonctions dont la phase et la fréquence varient dans le temps. Ces fonctions s'écrivent :

$$f(t) = A \cdot \cos[2 \cdot \pi \cdot f(t) + \varphi(t) + \varphi_0]$$

Le fonctionnement des synthétiseurs de fréquence repose sur le principe des asservissements, donc sur un système à boucle fermée qui permet la génération d'une grandeur de sortie quelconque (fréquence dans ce cas) à partir d'une grandeur d'entrée : la grandeur de référence.

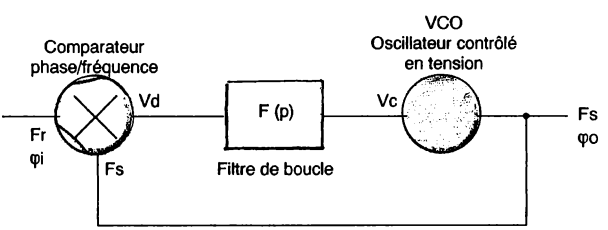



Fig. 8.1
Schéma de principe d'un PLL.

Le schéma d'une boucle élémentaire est représenté à la figure 8.1. Il consiste en :
- un détecteur ou **comparateur de phase/fréquence**,
- un **filtre de boucle** : filtre passe-bas,
- un oscillateur contrôlé par une tension : VCO.

La fonction du VCO est de générer une fréquence dont la valeur est donnée par la relation :

$$f(t) = f_0 + K_0 \cdot V_c(t)$$

Le signal de référence (grandeur de référence) a une fréquence $F_r$ et une phase $\varphi_i(t)$, et le VCO une fréquence $F_s$ et une phase $\varphi_0(t)$. Si nous admettons que le système est parfaitement linéaire, ce qui n'est jamais le cas du VCO, et que la boucle soit verrouillée, la tension de sortie du comparateur vaut :

$$V_d = K_D[\varphi_i(t) - \varphi_0(t)]$$

où $K_D$ est appelé facteur de **gain du comparateur** de phase et exprimé en volts par radian. La tension d'erreur $V_d$ est filtrée, le bruit et les composantes de fréquences élevées sont atténués. La fonction de transfert du filtre vaut $F(p)$.

La fréquence d'oscillation du VCO est fonction de la tension de contrôle qui lui est appliquée, la déviation de fréquence par rapport à la fréquence centrale vaut :

$$\delta f = K_0 \cdot V_c$$

où $K_0$est le **gain du VCO** exprimé en radian par seconde par volt, la fréquence dérivant de la phase. La fréquence instantanée est la dérivée de la phase instantanée.

$$\frac{d\varphi_0(t)}{dt} = K_0 \cdot V_c$$

En calcul opérationnel :

pour le VCO : $\varphi_0(p) = K_0 \cdot V_c(p)/p$

pour le comparateur de phase : $V_d(p) = K_D[\varphi_i(p) - \varphi_0(p)]$

et pour le filtre de boucle : $V_c(p) = F(p) \cdot V_d(p)$

La fonction de transfert en boucle fermée, $H(p)$, est obtenue en combinant ces trois équations :

$$H(p) = \frac{\varphi_0(p)}{\varphi_i(p)} \qquad H(p) = \frac{K_0 K_D F(p)}{p + K_0 K_D F(p)}$$

Dans le cas de l'introduction d'un diviseur par $N$ dans la chaîne de retour, entre le VCO et le comparateur de phase (voir figure 8.2), les équations du comparateur de phase et du VCO ne changent pas mais la boucle n'est plus à retour unitaire. Si on considère le système verrouillé :

$$\frac{F_s}{N} = F_r$$

Le comportement de cette boucle est défini par la fonction de transfert :

$$H(p) = \frac{K_0 K_D F(p)}{Np + (K_0 K_D F(p))}$$

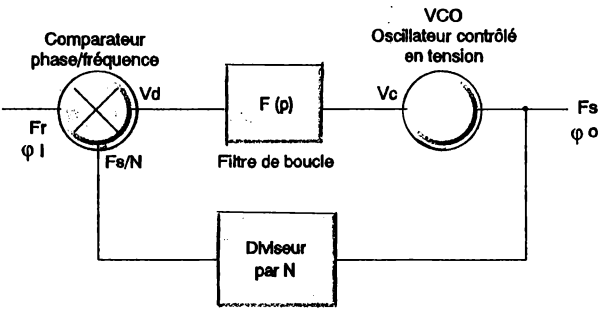



Fig. 8.2
Schéma de principe d'un PLL, introduction d'un diviseur par *N*.

Ce résultat reste valable quelle que soit la technique utilisée (voir figure 8.3), prédiviseur fixe par M suivi d'un diviseur programmable par P qui donne :

$$N = M\,P$$

ou prédiviseur à double module M, M + 1 géré par les compteurs A et B avec :

$$N = A + M\,B$$

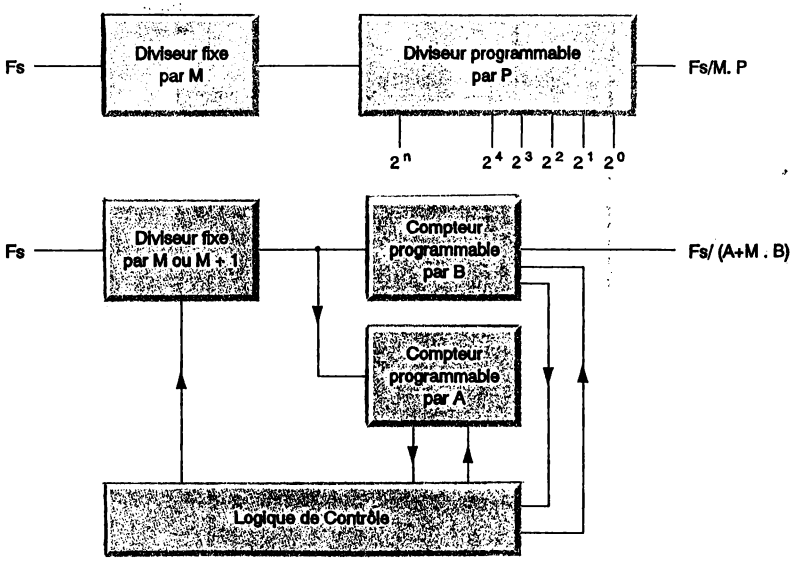



Fig. 8.3
Deux procédures pour diviser la fréquence de sortie du VCO.

Dans le cas des synthétiseurs à boucles multiples des figures 8.4 et 8.5, le résultat est presque aussi simple :

$$N = M_2 N_2/N_3 + M_3/P_3$$

Lorsque le système est verrouillé, les signaux présents à l'entrée du comparateur de phase sont de même fréquence. La fréquence du VCO vaut dans les quatre cas suivants :

$$F_s = M P F_R$$

$$F_s = (A + MB) F_R$$

et pour les synthétiseurs à deux et trois boucles des figures 8.4 et 8.5.

$$F_s = (M_1/N + M_2/N + 1) F_R$$

$$F_s = (M_1/N_1 + M_2/N_2 \times P_2 + M_3/N_3 \times P_2 \times P_3) F_R$$

Le changement de la fréquence de sortie du VCO est obtenu par modification du code des diviseurs programmables, le pas $F_{STEP}$ est défini par la plus petite variation pouvant être obtenue. Dans le premier cas, la simplicité du schéma n'est acquise qu'au déterminant du pas :

$$F_{STEP} = M \times F_R$$

que l'on peut comparer à $F_R$ pour la boucle fonctionnant avec un compteur à module double.

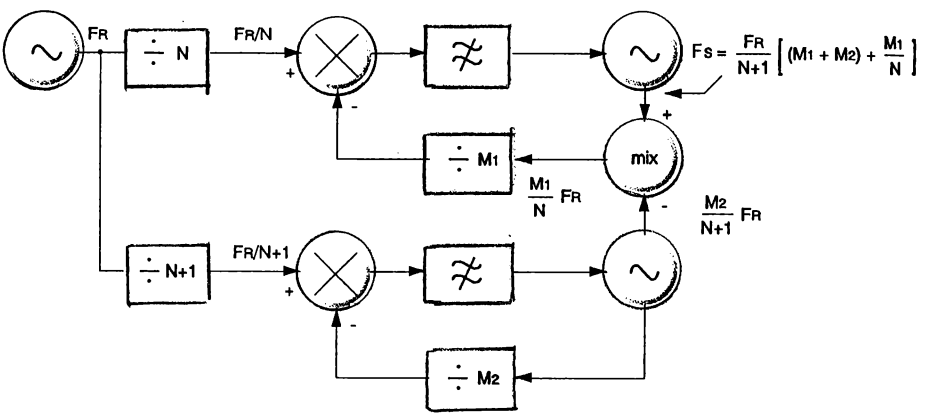



Fig. 8.4
Synthétiseur à double boucle.

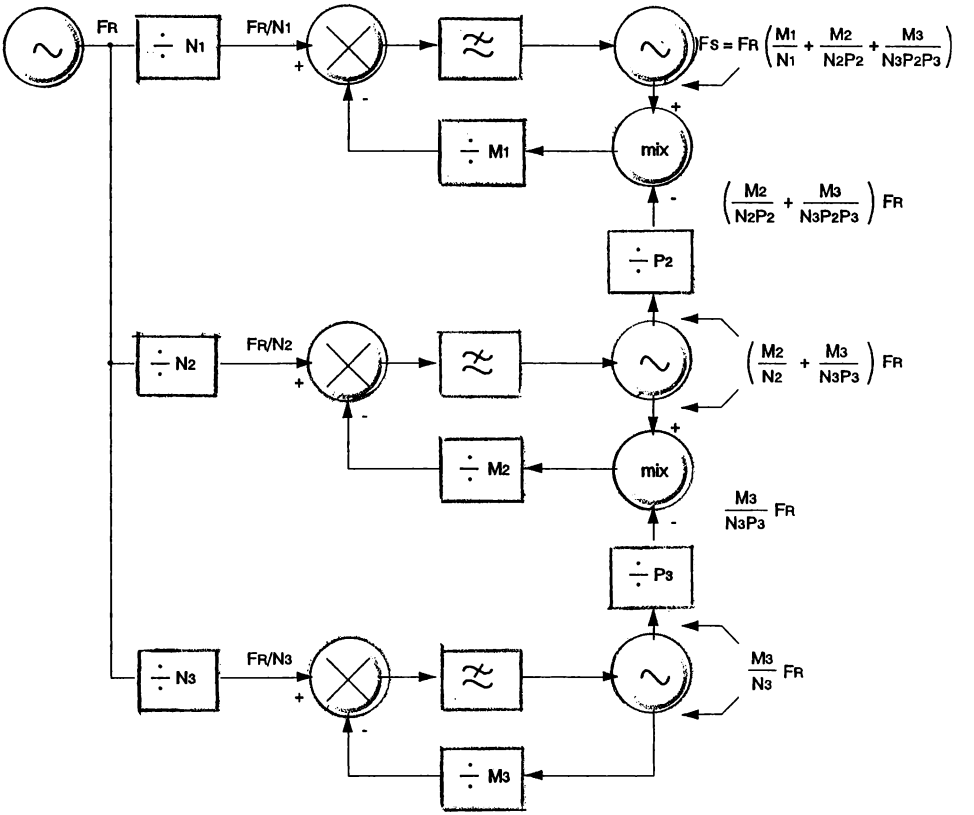



Fig. 8.5
Synthétiseur à triple boucle.

Des incréments très petits peuvent être obtenus avec les boucles multiples, pour la boucle de la figure 8.4. Si $M_1 + M_2$ est constant, le changement de $M_1$ en $M_1 + 1$ augmente la fréquence de sortie de $1/N(N + 1)F_R$. Pour une boucle triple comme celle de la figure 8.5, où l'on considère que chaque mélangeur met en évidence la différence des signaux reçus, l'incrément $F_{STEP}$ vaut :

$$\frac{F_R}{NP_2P_3}$$

Si les trois comparateurs de phase reçoivent à travers un diviseur par $N$ le même signal de référence :

$$F_{STEP} = \frac{F_R}{N}\left(M_1 + \frac{M_2}{P_2} + \frac{M_3}{P_2P_3}\right)$$

Ce type de boucle présente de nombreux avantages, l'adjonction de boucles élémentaires permet la réduction du pas sans changement de la fréquence de comparaison. Toutes les boucles élémentaires peuvent être construites suivant le même principe.

Il y a aussi quelques inconvénients puisque les mélangeurs élaborent de nombreux produits d'intermodulation indésirables, en plus du produit intéressant.

Ceci constitue le premier résultat important. Il apparaît clairement qu'avec un PLL, il est possible de générer un signal de fréquence quelconque ayant la même précision et la même stabilité que le signal à la fréquence de référence. La source de référence sera en général un oscillateur à quartz :

$$\frac{\Delta F_s}{F_s} = \frac{\Delta F_R}{F_R}$$

## Le filtre de boucle

Le choix des paramètres du filtre est un élément déterminant le fonctionnement et les caractéristiques du synthétiseur de fréquence : rapidité de réponse et amortissement. Divers filtres de boucle peuvent être utilisés. Par leur simplicité, les filtres passifs sont intéressants mais seul le **filtre à avance de phase**, en figure 8.6, peut dans la pratique être employé.

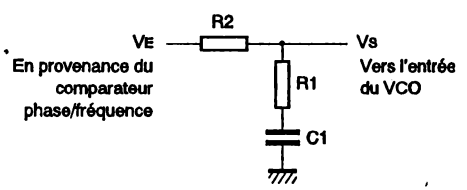



Fig. 8.6
Filtre à avance de phase.

Les filtres passe-bas classiques donnent de mauvais résultats quant à l'amortissement du système. On s'intéressera, plus particulièrement, aux filtres actifs : intégrateur et intégrateur suivi d'un passe-bas qui donnent respectivement un système d'ordre 2 et un système d'ordre 3. Pour des raisons de stabilité, devenant difficile à obtenir, l'ordre 3 est rarement dépassé. Toutefois il faut noter que les éléments parasites du circuit engendrent un filtrage supplémentaire et que l'ordre réel est toujours supérieur à l'ordre théorique.

La fonction de transfert du filtre précédent, figure 8.6, vaut :

$$F(p) = \frac{R_1 C_1 p + 1}{(R_1 + R_2) C_1 p + 1}$$

et la fonction du système bouclé :

$$H(p) = \frac{R_1 C_1 p + 1}{N\left(\frac{R_1 + R_2}{K_0 K_D}\right) C_1 p^2 + \left(\frac{N}{K_0 K_D} R_1 C_1\right) p + 1}$$

Avec $\omega_0$ représentant la **pulsation propre de la boucle** et $m$ l'**amortissement**, cette fonction peut être mise sous la forme :

$$H(p) = \frac{2mp/\omega_0 + 1}{p^2/\omega_0^2 + 2mp/\omega_0 + 1}$$

et dans ces conditions :

$$\omega_0 = \left(\frac{K_0 K_D}{N(R_1 + R_2)C_1}\right)^{1/2}$$

et

$$m = \frac{R_1 C_1 + N/K_0 K_D}{2}\left(\frac{K_0 K_D}{N(R_1 + R_2)C_1}\right)$$

$\omega_0$ et $m$ peuvent être choisis indépendamment l'un de l'autre en déterminant $R_1 + R_2$ pour un $\omega_0$ donné puis $m$ ajusté en conservant $R_1 + R_2$ constant et en faisant varier $R_1$. Le choix de $m$ et $\omega_0$ est une affaire de compromis et il n'existe pas d'optimisation systématique. L'analyse de la fonction de transfert permet d'étudier le comportement de la boucle dans le cas d'un **saut de phase** ou d'un **saut de fréquence**.

## 1 Saut de phase

Ceci se résume à la recherche de la réponse à un échelon unité de la fonction de transfert en $p$.

On recherche l'original de la fonction :

$$H_1(p) = \frac{1}{p} H(p)$$

Les calculs sont relativement simples mais un peu longs pour être détaillés dans cet ouvrage. Le lecteur pourra à titre d'exercice retrouver les résultats suivants :

$m < 1$

$$H_1(p) = 1 - e^{-m\omega_0 t}\left[\cos\left(\sqrt{1-m^2}\,\omega_0 t\right) - \frac{m}{\sqrt{1-m^2}}\sin\left(\sqrt{1-m^2}\,\omega_0 t\right)\right]$$

$m = 1$

$$H_1(p) = 1 + (\omega_0 t - 1)e^{-\omega_0 t}$$

$m > 1$

$$H_1(t) = 1 - e^{-m\omega_0 t}\left[ch\left(\sqrt{m^2-1}\,\omega_0 t\right) - \frac{m}{\sqrt{m^2-1}}sh\left(\sqrt{m^2-1}\,\omega_0 t\right)\right]$$

Les courbes de la figure 8.7 représentent la réponse de la boucle pour diverses valeurs du coefficient d'amortissement $m$. Dans la pratique, des valeurs comprises entre 0,5 et 1 donnent des résultats satisfaisants.

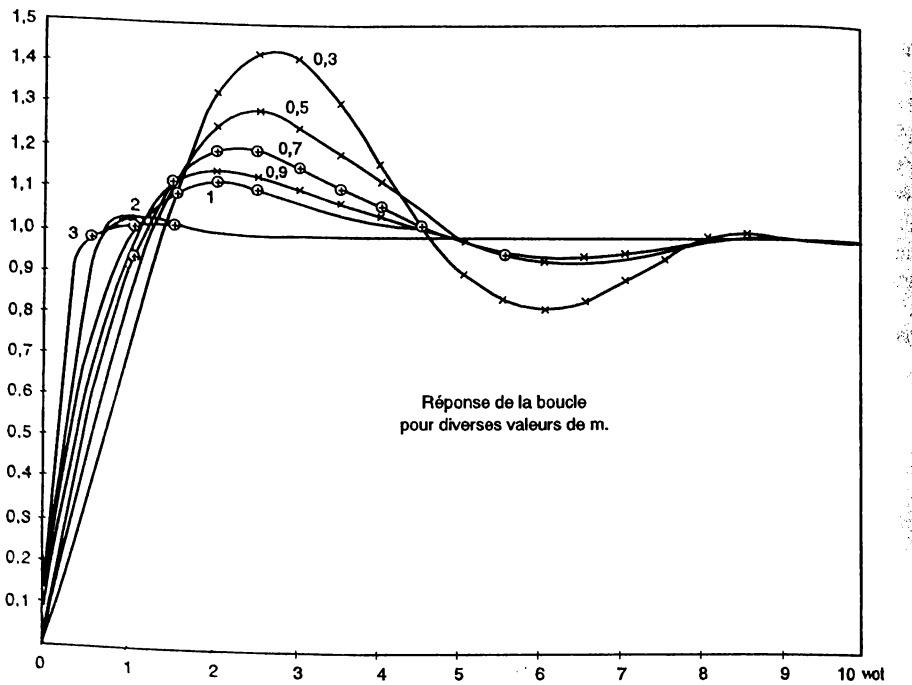



Fig. 8.7
Réponse du PLL pour diverses valeurs de l'amortissement.

## 2 Saut de fréquence

Ceci se résume à la recherche de la réponse à un échelon unité de la fonction de transfert en $p$.

On recherche l'original de la fonction :

$$H_2(p) = \frac{1}{p^2} H(p)$$

Le lecteur pourra à titre d'exercice retrouver les résultats.

Le système asservi doit posséder deux qualités fondamentales : la précision et la stabilité. Pour minimiser les effets du bruit en sortie, la bande passante de la boucle doit être aussi faible que possible. Pour minimiser l'erreur transitoire due à la modulation du signal ou pour obtenir les meilleures performances en ce qui concerne l'acquisition de la poursuite, la bande passante de la boucle doit être aussi large que possible. Ces deux principes sont directement opposés et si l'on améliore le comportement de la boucle en réduisant les effets du bruit, ce ne pourra être qu'au détriment de la vitesse d'acquisition et de la poursuite.

Pour ces notions, on se référera au paragraphe relatif à la plage de verrouillage et plage de capture.

On a, pour le filtre intégrateur de la figure 8.8, la fonction de transfert suivante :

$$F(p) = \frac{R_1 C_1 p + 1}{R_3 C_1 p}$$

Le système bouclé devient alors :

$$H(p) = \frac{R_1 C_1 p + 1}{\frac{N}{K_0 K_D} \times R_3 C_1 p^2 + R_1 C_1 p + 1}$$

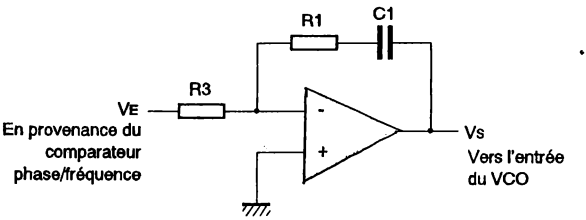



Fig. 8.8
Filtre actif donnant un système d'ordre 2.

Comme précédemment, on définit la pulsation propre de la boucle $\omega_0$ et le facteur d'amortissement $m$ :

$$\omega_0 = \sqrt{\frac{K_0 K_D}{N R_3 C_1}} \quad \text{et} \quad m = \frac{R_1 C_1}{2}\left(\sqrt{\frac{K_0 K_D}{N R_3 C_1}}\right)$$

Et il est encore plus aisé de faire varier $m$ par le biais de $R_1$, $R_3$ et $C_1$ fixant $\omega_0$ choisi préalablement. Les courbes de la figure 8.7 s'appliquent aussi à cette configuration.

Pour l'ordre 3 avec le filtre de la figure 8.9, l'équation est à peine plus complexe :

$$F(p) = \left(\frac{R_1 C_1 p + 1}{R_3 C_1 p}\right)\left(\frac{1}{R_2 C_2 p + 1}\right)$$

et la fonction de transfert du système bouclé :

$$H(p) = \frac{R_1 C_1 p + 1}{\left(\frac{N R_2 R_3 C_1 C_2}{K_0 K_D} p^3\right) + \left(\frac{N R_3 C_1}{K_0 K_D}\right) p^2 + R_1 C_1 p + 1}$$

La solution suivante donne de bons résultats confirmés par l'expérience :

$$H(p) = \frac{3p/\omega_0 + 1}{p^3/\omega_0^3 + 3p^2/\omega_0^2 + 3p/\omega_0 + 1} = \frac{3p/\omega_0 + 1}{(p/\omega_0 + 1)^3}$$

Ce qui donne :

$$R_1 C_1 = 3/\omega_0$$

$$R_3 C_1 = 3K_0 K_D / N\omega_0^2$$

$$R_2 C_2 = 1/3\omega_0$$

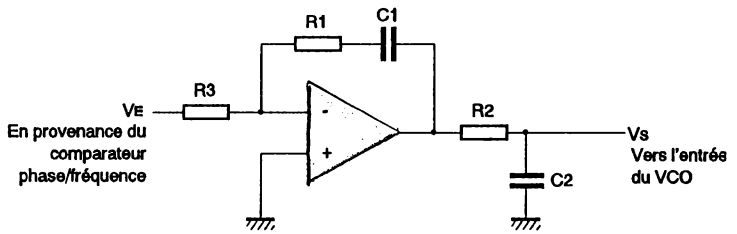



Fig. 8.9.
Filtre actif donnant un système d'ordre 3.

Si l'on s'intéresse à la réponse de la boucle à un saut de phase, on cherche l'original de la fonction :

$$H_1(p) = \frac{1}{p} H(p)$$

qui donne :

$$H_1(t) = [\omega_0^2 t^2 - \omega_0 t - 1]e^{-m\omega_0 t}$$

La figure 8.10 représente la réponse de la boucle.

On utilise très souvent le produit $K_0 K_D$. Les équations restent homogènes si l'on chiffre $K_0$ en Hz/V et $K_D$ en V/cycle qui sont les unités les plus accessibles aux mesures. Les trois filtres précédents ne peuvent être utilisés que par un comparateur de phase à sortie unique. Certains comparateurs possèdent deux sorties ne pouvant être actives simultanément, l'une pour $\varphi_i > \varphi_0$ et l'autre pour $\varphi_0 > \varphi_i$.

L'emploi des filtres des figures 8.11 et 8.12 est alors approprié et judicieux. Le filtre de la figure 8.11 a exactement la même fonction de transfert que le filtre de la figure 8.8, un calcul simple montre que les fonctions de transfert sont, au signe près, les mêmes pour les deux entrées.

Pour l'ordre 3, on trouve assez souvent le filtre de la figure 8.12 dont la fonction de transfert vaut :

$$F(p)=\left(\frac{R_3}{R_2+R_3}\right)\left(\frac{R_1C_1p+1}{R_3C_1p}\right)\left(\frac{1}{\frac{R_2R_3}{R_2+R_3}C_2p+1}\right)$$

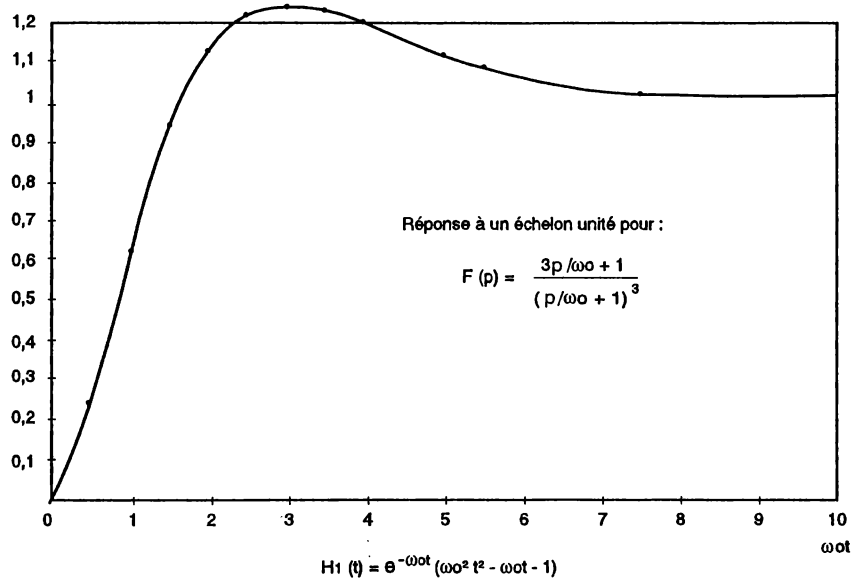



Fig. 8.10

Réponse de la boucle à un saut de phase.

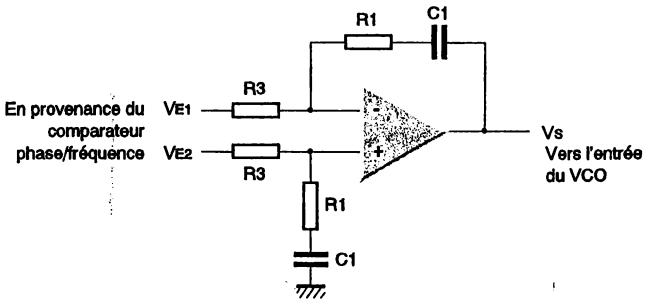



Fig. 8.11

Filtre actif donnant un système d'ordre 2 pour comparateur de phase ayant deux sorties.

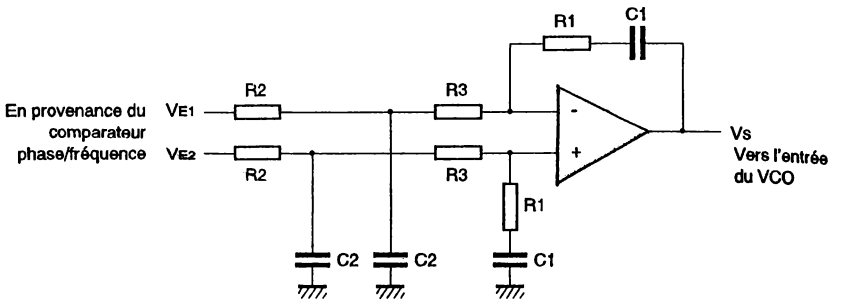



Fig. 8.12
Filtre actif donnant un système d'ordre 2 pour comparateur de phase ayant deux sorties.

## Plage de capture et plage de verrouillage

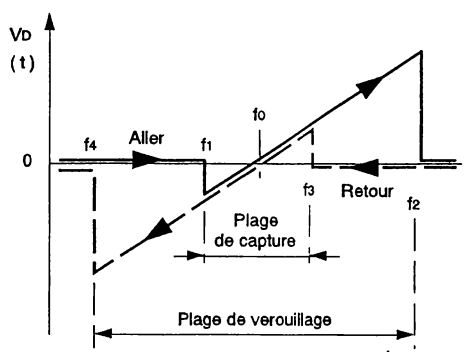



Fig. 8.13
Réponse du système à un signal d'entrée modulée (aller et retour).

Considérons le PLL à retour unitaire de la figure 8.1. Supposons que la tension d'entrée soit une rampe de fréquence. Le diagramme de la figure 8.13 représente la tension d'erreur en fonction de la fréquence d'entrée ; la plage de capture est limitée par les fréquences $F_1$ et $F_3$, et la plage de verrouillage par les fréquences $F_4$ et $F_2$.

La plage de verrouillage définit l'étendue de fréquence au voisinage de $F_0$ dans laquelle le PLL peut maintenir le verrouillage avec le signal d'entrée. Cette plage est aussi appelée **plage de maintien** et elle augmente en même temps que le gain de boucle.

La plage de capture définit les étendues de fréquence au voisinage de $F_0$ pour lesquelles le système peut établir ou acquérir le verrouillage avec le signal d'entrée. Elle est aussi appelée **plage d'acquisition** et elle est toujours plus petite que la plage de verrouillage. Elle est en étroite relation avec la bande passante du filtre de boucle. Elle décroît en même temps que la bande passante du filtre diminue.

À la figure 8.13, lorsque la fréquence augmente progressivement, la boucle ne répond au signal d'entrée que lorsque sa fréquence est supérieure à une limite inférieure $F_1$ correspondant à la limite inférieure de la plage de capture. La boucle se verrouille alors sur le signal d'entrée en créant une tension d'erreur. Puis la tension d'erreur varie avec la fréquence en suivant une pente égale au

facteur de conversion du VCO. La poursuite fonctionne jusqu'à ce que la fréquence atteigne $F_2$, correspondant à la limite maximale de la plage de verrouillage.

Lorsque la fréquence décroît, le cycle est modifié, la boucle est asservie à partir de $F_3$ et la poursuite fonctionne jusqu'à $F_4$.

## Modulation d'une boucle à verrouillage de phase

Le PLL peut être modulé en phase ou en fréquence. Le schéma de la figure 8.14 rend compte de l'introduction des signaux modulants dans le PLL.

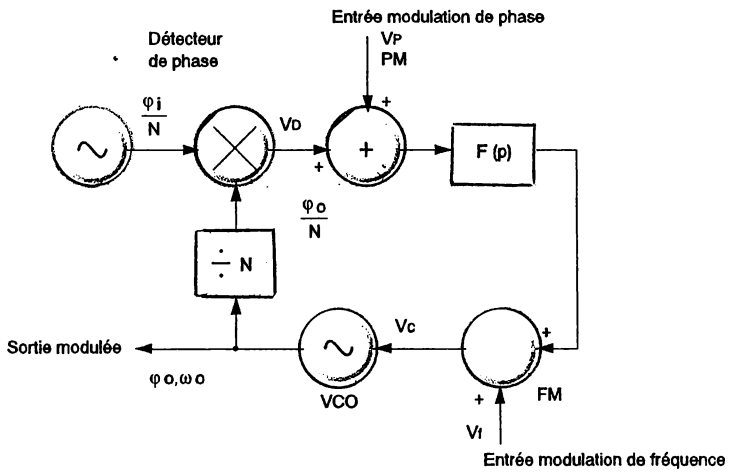



Fig. 8.14
Modulation de phase et de fréquence dans un PLL.

### Modulation de phase

Par principe, le système bouclé tend à annuler la tension $V_p + V_d$, et ceci n'est possible que si une erreur de phase crée une tension $V_d$ qui annule $V_p$. Si le système génère une tension correctrice, c'est qu'il existe un décalage de phase entre la phase du VCO et la phase du générateur de référence. Le but, qui est la modulation de phase, est atteint. Et en utilisant les mêmes notations que précédemment :

$$\varphi_0(p) = \frac{H(p) \cdot V_p(p)}{K_D}$$

La sensibilité – en modulation – du circuit vaut $1/K_D \cdot H(p)$. La fonction $H(p)$ étant la fonction de transfert d'un filtre passe-bas, la fréquence de coupure de la boucle devra être beaucoup plus grande que la plus grande fréquence modulante. Pour éviter les distorsions, la caractéristique du détecteur de phase doit être linéaire, celle du VCO est moins critique puisque les non-linéarités sont réduites par l'effet de contre-réaction de la boucle.

## 2 Modulation de fréquence

En additionnant le signal modulant $V_f$ au signal de sortie du filtre, on est dans le cas de la modulation de fréquence et on obtient :

$$\varphi_0(p) = \frac{K_0[1 - H(p)]}{p} V_f(p)$$

La fréquence étant la dérivée de la phase :

$$\omega_0(t) = \frac{d\varphi_0(t)}{dt}$$

ce qui donne :

$$\omega_0(p) = p\,\varphi_0(p)$$

et donc

$$\omega_0(p) = K_0[1 - H(p)]V(p)$$

Mais $[1 - H(p)]$ est la fonction de transfert d'un filtre passe-haut, la fréquence de coupure basse doit donc être plus faible que le signal de plus basse fréquence à transmettre. Le modulateur de fréquence basé sur un PLL ne peut donner un décalage constant (voir figure 8.15).

Contrairement à la modulation de phase, la linéarité du VCO est capitale, et la non linéarité du comparateur de phase est tolérable puisque réduite par l'effet de contre-réaction de la boucle. Une boucle à verrouillage de phase PLL peut être modulée en fréquence par un signal à condition que celui-ci ne possède pas d'énergie dans la bande comprise entre 0 et $\omega_0$ réservée à l'asservissement.

Ceci signifie qu'un signal numérique NRZ ne peut pas être appliqué à l'entrée du VCO pour obtenir une modulation de fréquence. La densité spectrale de puissance d'un signal NRZ – voir chapitre 9 sur les modulations numériques – contient de l'énergie à la fréquence nulle. A contrario, un signal analogique compris par exemple en 300 Hz et 3 400 Hz peut moduler un PLL à condition que $\omega_0$ soit très inférieur à $2\pi \times 300$ rd/s.

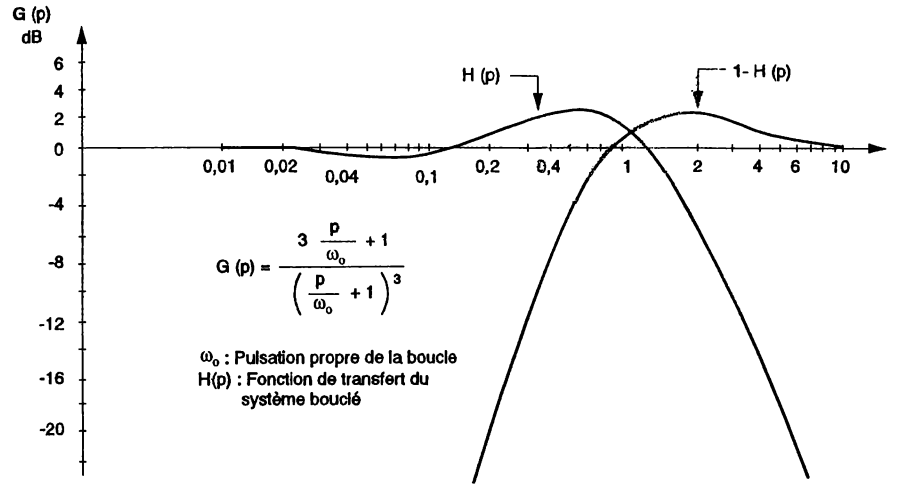



Fig. 8.15
Réponse d'un PLL modulé en fréquence.

Dans une application pratique, on prendra :

$$\omega_0 = 2\pi F_0 \quad \text{avec} \quad F_0 << \frac{F_{min}}{10}$$

$F_{min}$ est la fréquence minimale du signal modulant.

## Démodulation par une boucle à verrouillage de phase

On peut démontrer que le PLL est le démodulateur de fréquence optimal, et qu'il peut être utilisé comme tel (figure 8.16).

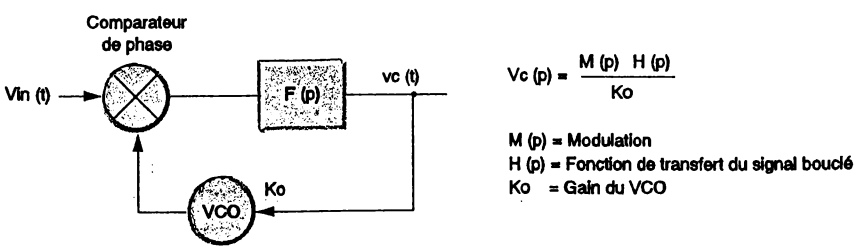



Fig. 8.16
PLL utilisé en démodulateur de fréquence.

On suppose qu'un signal modulé en fréquence est injecté à l'entrée du comparateur de phase. Pour que la boucle reste verrouillée, il est nécessaire que la fréquence du VCO suive exactement la fréquence du signal incident. La tension de contrôle est donc une réplique exacte du signal modulant la tension d'entrée. Pour le signal $V_c(t)$, la fonction de transfert vaut :

$$V_c(p) = \frac{M(p)\,H(p)}{K_0}$$

où $M(p)$ représente la modulation, $H(p)$ la fonction de transfert du système bouclé et $K_0$ le gain du VCO.

Le signal démodulé est donc le signal d'origine filtré par la fonction de transfert du système bouclé. Si le signal modulant a un spectre compris entre $F_{min}$ et $F_{max}$, la pulsation naturelle de la boucle $\omega_o$ sera choisie supérieure à $F_{max}$. Le démodulateur à PLL ne peut donc fonctionner qu'avec des fréquences de comparaison, $F_{VCO}$, élevées :

$$F_{max} < F_0 << F_{VCO}.$$

Pour cette configuration, un **modulateur en anneau** peut être employé en tant que comparateur de phase. La figure 8.17 montre la structure interne du comparateur de phase qui sera examiné ultérieurement.

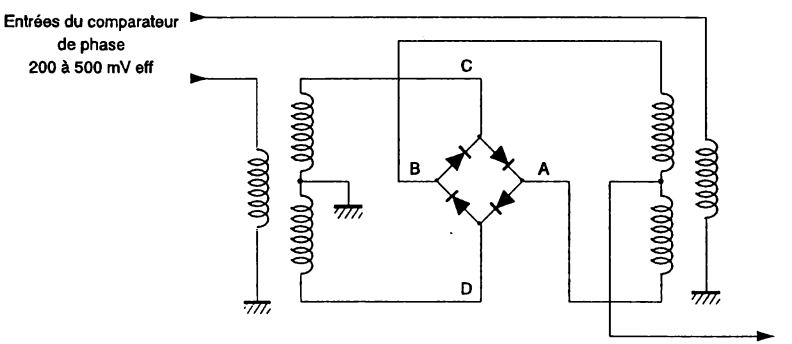



Fig. 8.17
Modulateur en anneau utilisé en tant que comparateur de phase.

## Bruit de phase

Pour la définition et la mesure de **bruit de phase**, on se reportera à la figure 8.18. Le bruit de phase concerne une porteuse pure. Un signal sinusoïdal est mathématiquement défini par :

$$V(t) = A \cdot \sin(\omega t + \varphi)$$

Ce signal est pur, ce qui signifie qu'il n'est pas modulé en amplitude, action sur $A$, ni en fréquence, action sur $\omega$, ni en phase.

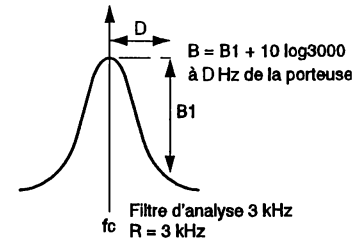



Fig. 8.18
Définition et mesure du bruit de phase.

action sur $\varphi$. La figure montre ce que l'on visualise avec un analyseur de spectre. Le bruit de phase est donné par la relation :

$$B = B_1 + 10\log R$$

Dans cette relation, $B_1$ est la distance entre le niveau crête et le plancher de bruit à une distance donnée de la porteuse, et $R$ est la largeur du filtre d'analyse.

Plus la valeur $B$ est importante, plus la qualité du PLL est bonne, en termes de bruit de phase. Attention, car on mesure simultanément le bruit de phase du quadripôle en test et celui de l'appareil de mesure. On se trouve exactement dans le même cas que celui de la mesure du temps de montée avec un oscilloscope. La figure 8.18 est relative à un oscillateur très stable comme peut l'être un oscillateur à quartz ou un VCO dont la mesure est faite au vol sans boucle de stabilisation. La figure 8.18 montre ce que devient le bruit de phase lorsque le VCO est stabilisé par la boucle. Ce que l'on peut appeler la remontée du plancher de bruit au voisinage immédiat de la porteuse est fonction de la valeur du diviseur. Ce résultat est très intéressant car il montre le compromis qui existe entre le bruit de phase au voisinage immédiat de la porteuse et le plus petit pas de fréquence. Le bruit de phase va donc perturber les modulations de fréquence et de phase. Plus la largeur occupée par le bruit est faible par rapport à la largeur occupée par le signal, moins celui-ci est gênant. Il est donc clair que le bruit de phase sera critique dans les applications à bande étroite ou très étroite.

Dans le cas d'un PLL, le bruit de phase est fonction, bien sûr, de la valeur du diviseur mais aussi du VCO lui-même, du filtre de boucle et de la réalisation pratique. Si l'on cherche un bruit de phase faible, il faut soigner les alimentations, les découplages et employer des amplificateurs à faible bruit pour le filtre.

Toutes ces actions n'ont qu'un seul but : éviter tout bruit parasite sur la tension de commande.

## Éléments constituant une boucle à verrouillage de phase

Dans ce paragraphe, nous nous intéresserons uniquement aux oscillateurs contrôlés en tension, (VCO), aux comparateurs de phase et finalement aux oscillateurs de référence à quartz. Les

diviseurs programmables sont des circuits logiques traités dans le chapitre 2 relatif à la logique séquentielle.

## 1 VCO

Un oscillateur contrôlé en tension a une fonction de transfert du type de celle de la figure 8.19. Dans cet exemple, la fréquence de sortie croît avec la tension de contrôle. La tension d'entrée ou tension de commande est une tension continue donc dépourvue de composantes alternatives. Pour le signal de sortie, en fonction du type d'application, on peut rechercher soit un maximum de pureté spectrale et l'on est en présence d'une sinusoïde, soit une forme d'onde rectangulaire.

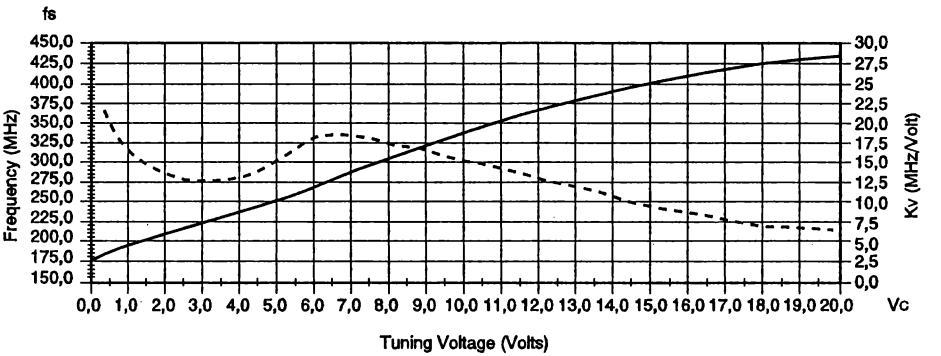



Fig. 8.19
Caractéristiques de transfert d'un VCO, $f_s = K_v \cdot V_c$.

Le rapport $\Delta f/\Delta V$ est appelé gain du VCO : $K_0$.
Le meilleur VCO serait défini par les caractéristiques suivantes :
- stabilité de phase, bruit de phase faible,
- bonne linéarité sur $K_0$,
- grand gain $K_0$,
- acceptation d'une modulation large bande.

Il est impossible de satisfaire simultanément les quatre critères précédents, les trois derniers points étant en opposition avec la stabilité et le bruit de phase.

Les oscillateurs contrôlés en tension comportent les trois catégories suivantes.

### *1.1 Oscillateurs à quartz VCXO*

Par construction, un quartz a un fort coefficient de surtension $Q$. En conséquence, il est assez difficile de décaler sa fréquence de résonance. Les VCXO ont donc un faible gain $K_0$.

En pratique, pour des VCXO dont la fréquence est comprise entre 5 et 20 MHz, le rapport $\Delta f/f_0$ est compris entre 100 ppm et 400 ppm.

Exemple : $f_0 = 20$ MHz.

Si l'excursion de fréquence vaut 400 ppm, lorsque la tension de commande évolue de 0,5 V à 4,5 V, on obtient le gain du VCO : $K_0 = 2\,000$ Hz/V.

Ce résultat pourra être comparé avec celui obtenu avec les oscillateurs LC dans les lignes qui suivent.

### *1.2 Oscillateurs LC*

Tout type d'oscillateur LC comme celui de la figure 8.20 peut être transformé en oscillateur contrôlé en tension. Cette transformation consiste à remplacer une ou plusieurs capacités fixes par une ou plusieurs **diodes à capacités variables** (voir la figure 8.21).

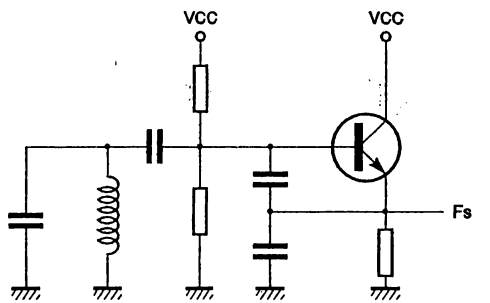



Fig. 8.20
Oscillateur LC fixe pouvant être transformé en VCO.

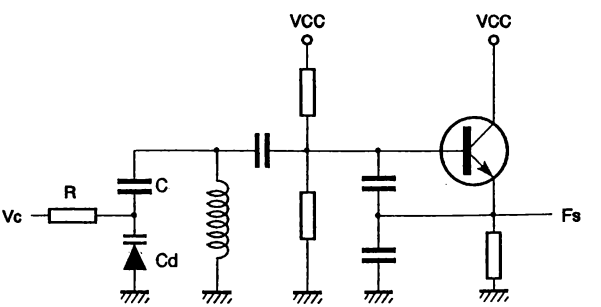



Fig. 8.21
Transformation en VCO.

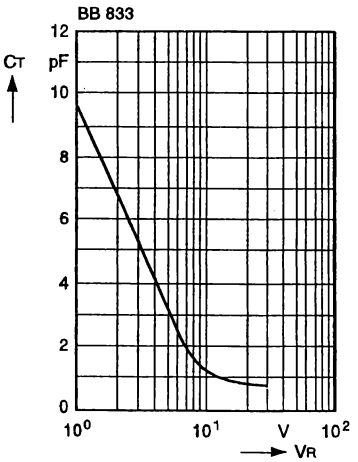



Fig. 8.22

Évolution de la capacité de la diode varicap en fonction de la tension interne appliquée à ses bornes.

La capacité de cette diode évolue de la manière indiquée par la courbe de la figure 8.22 en fonction de la tension **inverse** appliquée à ses bornes.

Pour la diode à capacité variable BB 833 prise en exemple, le rapport $C_{max} / C_{min}$ vaut 12.

Si l'on admet, mais ce n'est jamais le cas, que seule la capacité des diodes intervient dans la détermination de la fréquence d'oscillation :

$$f_{max} = \frac{1}{k\sqrt{LC_{min}}} \quad \text{et} \quad f_{min} = \frac{1}{k\sqrt{LC_{max}}}$$

et le rapport :

$$f_{max} / f_{min} = \sqrt{C_{max} / C_{min}}$$

Dans le cas de la diode BB 833 :

$$f_{max} / f_{min} = \sqrt{12} = 3{,}4$$

Dans la pratique, les diverses capacités fixes annexes et les capacités parasites limitent ce rapport à des valeurs inférieures. Des rapports $f_{max}/f_{min}$ de l'ordre de 2 sont courants et très rarement dépassés.

On peut donc imaginer un VCO dont les fréquences $f_{max}$ et $f_{min}$ seraient respectivement 1 000 MHz et 2 000 MHz pour une tension de commande évoluant de 0 à 20 V.

Le gain d'un tel VCO vaut :

$$K_o = 50 \text{ MHz/V}$$

Cette valeur est à comparer avec celle obtenue dans le cas du VCXO.

Au schéma de la figure 8.21, la résistance $R$ associée à la diode varicap constitue un filtre passe-bas dont la fréquence de coupure vaut :

$$f_c = \frac{1}{2\pi R C_{max}}$$

avec :

$$C_{max} = C_d + C$$

Cette fréquence de coupure est importante à deux titres : premièrement elle génère un pôle supplémentaire dans la fonction de transfert du système bouclé, et deuxièmement elle limite la bande d'un éventuel signal modulant.

Si par exemple le PLL doit être modulé par un signal en bande de base jusqu'à 5 MHz, la résistance $R$ maximale peut être calculée sans difficulté. Plus la fréquence de modulation doit être importante, plus la valeur de $R$ doit être faible.

En contrepartie, plus la résistance $R$ est faible, plus le coefficient de surtension du circuit oscillant est faible, et par conséquent plus le bruit de phase se dégrade.

On note finalement que la courbe $C = f(V_r)$ de la diode varicap est telle qu'il est impossible d'obtenir une bonne linéarité de $K_0$ sur une large plage de fréquence.

Toutefois pour un VCO donné entre $f$ et $2f$, il est couramment admis que la linéarité est suffisante autour de $f_0$ sur une plage de quelques pour-cent.

Ceci illustre parfaitement les difficultés du concepteur qui doit effectuer un compromis au moment du choix ou de la conception du VCO.

### *1.3 Multivibrateurs*

Le schéma de la figure 8.23 (page 260 et 261) donne un cas pratique de multivibrateur dont la fréquence de sortie est contrôlée par une tension d'entrée. Le principe de ces VCO repose toujours sur les charges et décharges successives d'une capacité. La tension de commande fixe un courant constant de charge, dès que la charge fixée est atteinte, la capacité est déchargée et un nouveau cycle recommence. Les multivibrateurs ont en général de pauvres caractéristiques en ce qui concerne le bruit de phase. Ce bruit est principalement dû à l'instabilité de la tension de seuil des comparateurs.

Pour une structure donnée, un multivibrateur fonctionne à une fréquence élevée à un bruit de phase moins important que ce même multivibrateur fonctionnant en basse fréquence.

En conséquence, dans la pratique, pour diminuer le bruit de phase, le PLL fonctionne à une fréquence multiple de la fréquence souhaitée et le signal utile est obtenu par divisions successives.

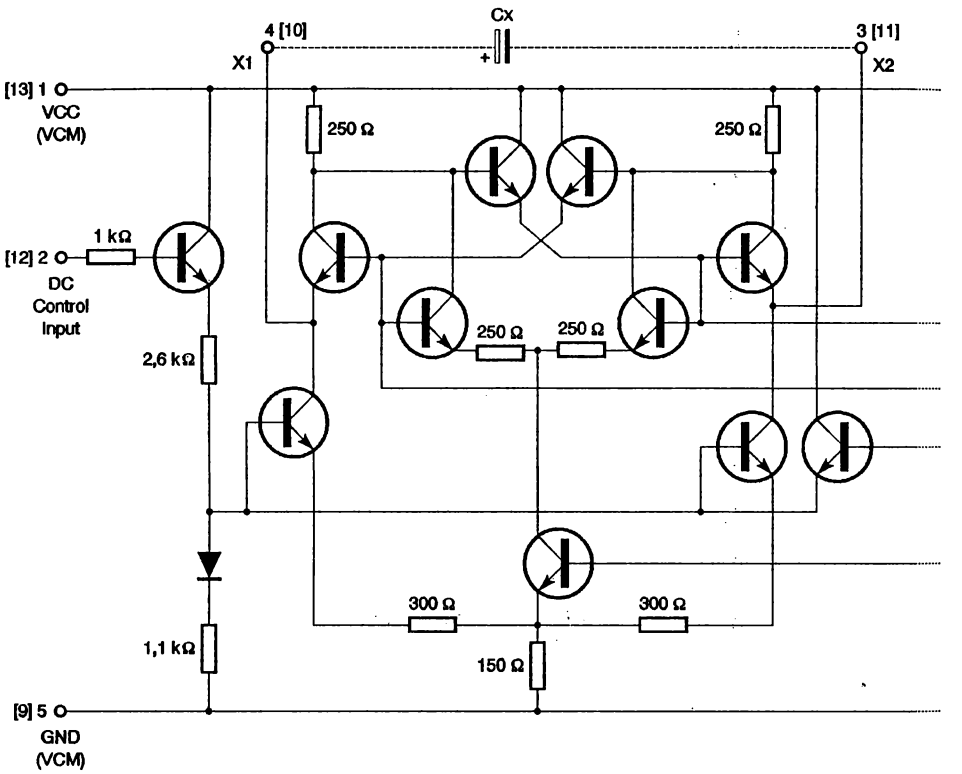



Fig. 8.23

Schéma de principe d'un multivibrateur contrôlé en tension VCM.

## 2 Détecteur de phase

Il existe deux catégories de détecteurs de phase : les circuits **multiplicateurs analogiques,** qui délivrent le produit de deux tensions d'entrée et les **circuits séquentiels** qui opèrent à partir des transitions sur des signaux d'entrée numériques.

### *2.1 Multiplicateurs analogiques*

Supposons être en présence d'un multiplicateur idéal auquel on injecte sur chacune des deux entrées :

$$v_0(t) = V_0 \cdot \sin(\omega t + \theta_0) \qquad \text{et} \qquad v_1(t) = V_1 \cdot \sin(\omega t + \theta_1)$$

La tension de sortie du multiplicateur vaut :

$$vd = Km \cdot v_0(t) \cdot v_1(t)$$

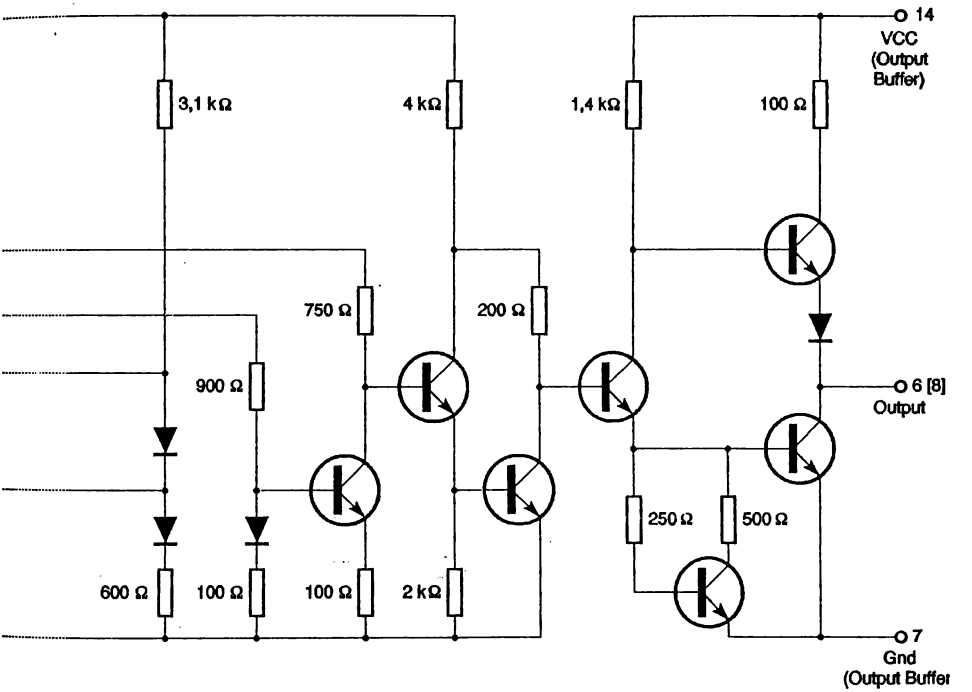



Fig. 8.23 (suite)

et si l'on pose :

$$Km \cdot V_0 \cdot V_1 = K_D$$

on obtient finalement le signal de sortie :

$$vd = K_D \cdot \sin(\theta_1 - \theta_0) + K_D \cdot \sin(2\omega t + \theta_0 + \theta_1)$$

Le terme $K_D \cdot \sin(2\omega t + \theta_0 + \theta_1)$ est une composante au double de la fréquence d'entrée. Cette composante devra impérativement être éliminée soit par le filtre de boucle soit par un filtre rejecteur auxiliaire. Après élimination de cette composante il subsiste :

$$vd = K_d \cdot \sin(\theta_1 - \theta_0)$$

Ce qui montre que le multiplicateur, dans ces conditions, délivre une information, fonction de la différence de phase des signaux d'entrée.

Pour ce type de comparateur de phase, les deux signaux d'entrée sont des signaux sinusoïdaux et le signal de sortie utile est une tension continue.

Si les signaux d'entrées étaient remplacés par des signaux comportant des harmoniques, des signaux carrées par exemple, des produits d'intermodulation apparaîtraient en sortie.

Un multiplicateur analogique, comme celui de la figure 8.24, peut être utilisé en guise de comparateur de phase. Cette structure est réservée aux circuits intégrés dans lesquels on peut espérer un bon appariement des transistors.

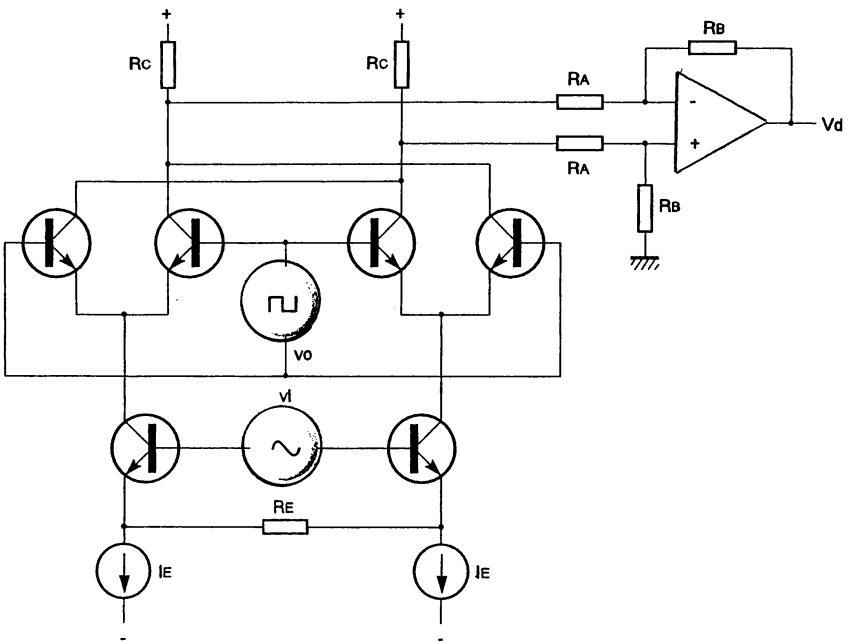



Fig. 8.24
Multiplicateur analogique utilisé en comparateur de phase.

Pour obtenir un comparateur de phase, un mélangeur équilibré, dont le schéma est donné à la figure 8.17, est un meilleur choix. Ce comparateur peut fonctionner jusqu'à des fréquences plus élevées, plusieurs centaines de MHz, que ne peuvent le faire les comparateurs de phase à transistors.

Comme pour le multiplicateur idéal, la tension d'erreur est de la forme :

$$vd = Vm \cdot \sin(\theta_1 - \theta_0)$$

La fonction de transfert est donnée à la figure 8.25. La tension d'erreur évolue entre – 0,3V et + 0,3V. Cette tension est en général inutilisable directement pour le pilotage de l'oscillateur contrôlé en tension.

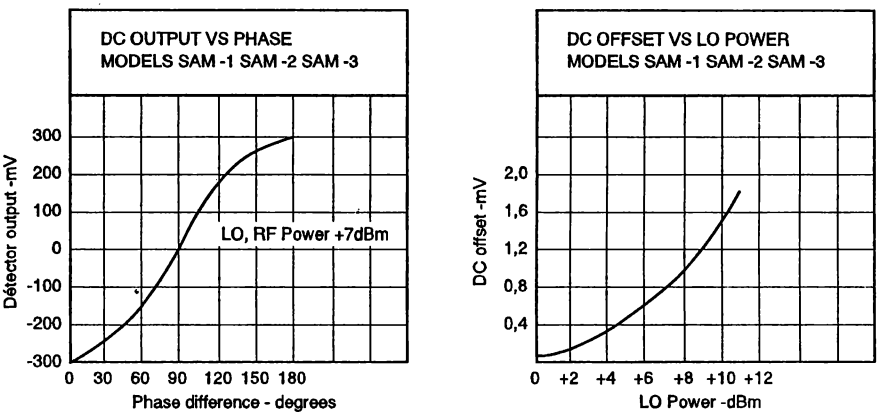



Fig. 8.25
Fonction de transfert du mélangeur équilibré.

Un amplificateur opérationnel sera intercalé entre le filtre de boucle et l'entrée de commande du VCO, conformément au schéma de la figure 8.26.

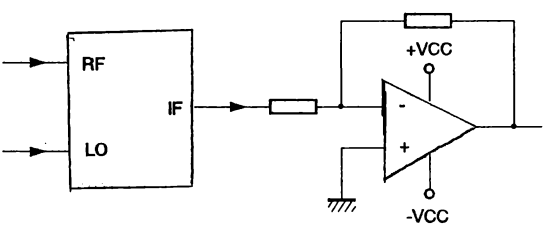



Fig. 8.26
Amplification du signal d'erreur en sortie du mélangeur équilibré.

Le mélangeur équilibré n'est pas un multiplicateur idéal, en conséquence la sortie est constituée de la tension d'erreur et de composantes multiples de la fréquence de fonctionnement. Ces composantes seront éliminées par filtrage.

En résumé, on retiendra que les multiplicateurs peuvent être utilisés pour la comparaison de signaux de phases différentes mais de fréquences **identiques**.

### *2.2 Circuits logiques séquentiels*

Ces circuits opèrent sur les fronts des signaux logiques d'entrée. Les signaux d'entrée sont des signaux logiques obtenus par division des fréquences de référence et fréquence du VCO.

Le plus simple de ces comparateurs de phase est une simple bascule RS. Les fronts descendants d'une des entrées positionnent la sortie au 1 logique et les fronts descendants sur le seconde entrée effectuent la remise à zéro.

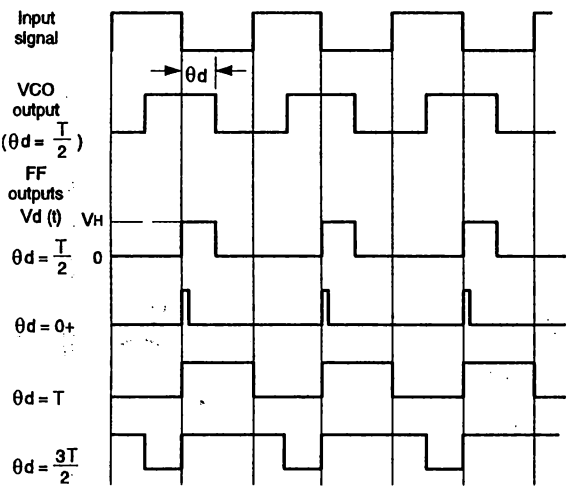



Fig. 8.27
Bascule RS utilisée en comparateur de phase, diagramme des temps.

Le diagramme des temps de la figure 8.27 donne l'allure des formes d'onde pour diverses valeurs de $\theta_d$ : on obtient un décalage de phase entre les deux entrées.

Pour $0 < \theta_d < 2\pi$ :

$$V_d = V_H \cdot \theta_d$$

Dans cette relation, $V_H$ est la tension du signal logique à l'état haut.

Ce circuit est simple mais rarement employé, on lui préfère une version légèrement plus complexe appelée **comparateur de phase/fréquence**.

La figure 8.28 représente la structure de ce type de comparateur qui peut aussi être fabriqué à partir de circuits logiques standards comme le montre la figure 8.29.

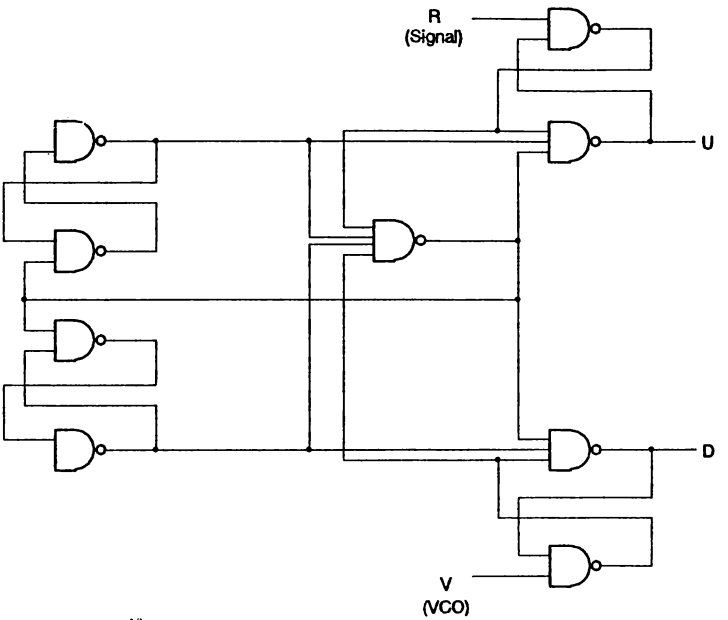



Fig. 8.28
Comparateur phase/fréquence.

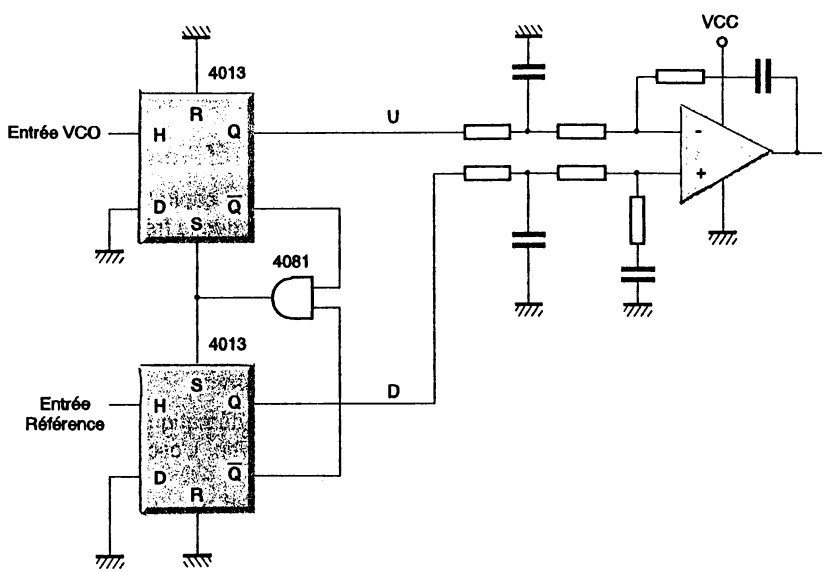



Fig. 8.29
Comparateur phase/fréquence en composants discrets.

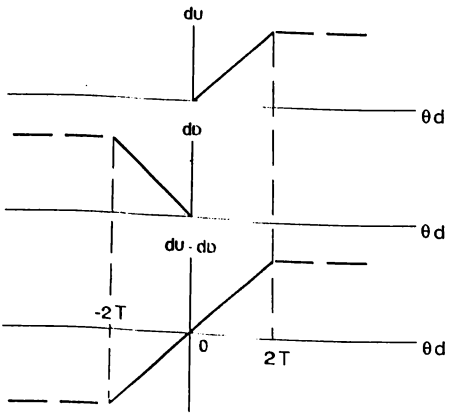



Fig. 8.30
Fonction de transfert du comparateur phase/fréquence.

Les sorties U et D sont actives à l'état bas. Les deux sorties ne peuvent être actives simultanément. La sortie U est active si la fréquence du VCO est en avance sur le signal de référence, la sortie D est active si la fréquence du VCO est en retard sur la référence.

La fonction de transfert de ce comparateur phase/fréquence (voir la figure 8.30), est apériodique, et à ce titre, est intéressante. Elle montre que si la boucle n'est pas verrouillée, une des sorties reste active et après filtrage, génère une rampe jusqu'à l'acquisition en phase de la boucle.

Ce type de comparateur phase/fréquence est largement le type le plus employé dans les circuits intégrés regroupant les diviseurs, le comparateur de phase et parfois l'oscillateur pour la fréquence de référence.

La fréquence de fonctionnement dépend directement de la technologie utilisée dans les circuits logiques. Avec des circuits CMOS, la fréquence sera inférieure à quelques centaines de kHz.

Avec des circuits HCMOS, la fréquence pourra atteindre plusieurs MHz. Grâce à la technologie BICMOS, la fréquence de fonctionnement peut atteindre 100 MHz – circuit QUALCOMM –.

Cette fréquence est une limite et comme il a été précisé précédemment, au-delà de cette limite, on aura recours à un mélangeur équilibré.

## 3 Oscillateurs à quartz de référence

L'oscillateur à quartz est un élément capital dans la constitution d'un synthétiseur de fréquence puisque la fréquence de sortie est un multiple de la fréquence de référence.

On a, par exemple, les deux relations suivantes, dans lesquelles $F_s$ est la fréquence de sortie et $F_r$ la fréquence de référence :

$$F_s = M \cdot P \cdot F_r$$

$$F_s = (A + MB) \cdot F_r$$

Toute instabilité ou dérive est automatiquement multipliée et répercutée sur la fréquence de sortie. L'oscillateur de référence doit donc être irréprochable.

Le schéma de l'oscillateur à quartz le plus répandu est donné à la figure 8.31. Cet oscillateur est bâti autour d'une porte inverseuse de type CMOS. Cet oscillateur est capable de fonctionner jusqu'à quelques dizaines de MHz avec des **quartz en mode fondamental** et résonance parallèle. En terme de bruit de phase, les

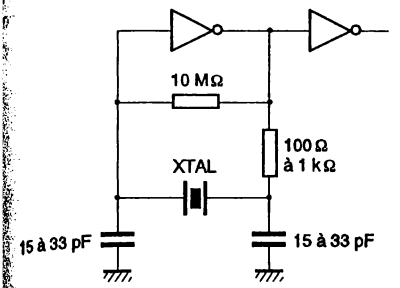



Fig. 8.31
Oscillateur à quartz autour d'une porte CMOS.

caractéristiques de cet oscillateur sont très mauvaises. Ce type de schéma, appliqué à un PLL, doit être banni dans la plupart des cas.

L'usage du schéma de la figure 8.31 doit être réservé aux horloges des microcontrôleurs ou microprocesseurs.

Pour des performances optimales, on optera pour un schéma d'oscillateur comme celui de la figure 8.32. Ce schéma d'oscillateur est utilisable avec des quartz en mode fondamental et résonance parallèle jusqu'à des fréquences de 20 MHz environ.

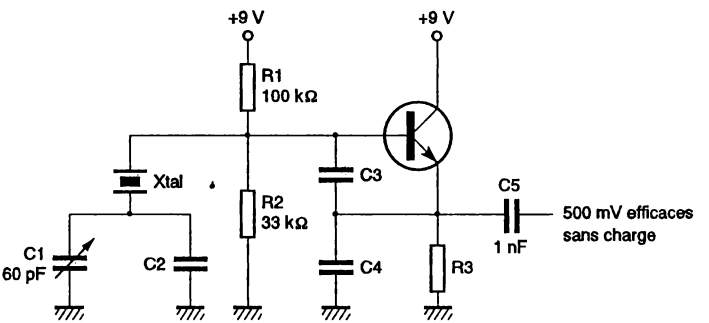



Fig. 8.32
Oscillateur pour quartz en mode fondamental jusqu'à 20 MHz.

Les valeurs à adopter pour le schéma de la figure 8.32 sont fonctions de la fréquence et données dans le tableau ci-dessous.

| Fréquence MHz | $R_3$ | $C_2$ pF | $C_3$ pF | $C_4$ pF |
|---|---|---|---|---|
| 1 à 3 | 3,3 K | aucune | 220 | 220 |
| 3 à 6 | 3,3 K | 33 | 150 | 150 |
| 6 à 10 | 2,2 K | 33 | 150 | 150 |
| 10 à 18 | 1,2 K | aucune | 100 | 100 |
| 18 à 20 | 680 | aucune | 68 | 33 |

Pour des fréquences supérieures, de 15 à 100 MHz, on peut adopter le schéma de la figure 8.33. Les quartz oscillent alors sur les partiels 3 ou 5 en résonance série.

Les quartz sont alors fabriqués spécialement pour vibrer dans ces modes.

Sur le schéma de la figure 8.33, lorsque le quartz est temporairement court-circuité, le circuit oscille et la fréquence d'oscillation doit être proche de la fréquence de résonance série du quartz. Cette fréquence est ajustée par $L_1$ et $C_1$.

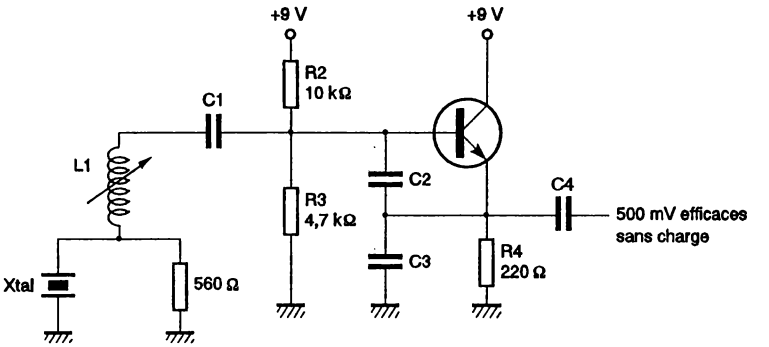



Fig. 8.33
Oscillateur à quartz partiel 3 ou 5 jusqu'à 100 MHz environ.

Les valeurs à adopter pour le schéma de la figure 8.33 sont fonction de la fréquence et des données dans le tableau ci-dessous.

| Fréquence MHz | $R_3$ | $C_2$ pF | $C_3$ pF | $C_4$ pF |
|---|---|---|---|---|
| 15 à 20 | 100 | 100 | 68 | 33 |
| 20 à 26 | 100 | 100 | 68 | 33 |
| 26 à 48 | 100 | 68 | 47 | 33 |
| 48 à 63 | 68 | 33 | 15 | 15 |

Chapitre 9

# MODEMS ET TRANSMISSIONS NUMÉRIQUES

## SOMMAIRE

- Milieu de transmission
- Problèmes liés au signal NRZ
- Transmission en bande de base
- Modulation autour d'une fréquence porteuse

## Définition du signal numérique

*Dans de nombreux cas, il est nécessaire de transmettre le code binaire d'une grandeur. Le signal se présente alors comme une succession de 0 et de 1. Dans le codage dit NRZ, ils correspondent à deux niveaux électriques donnés, l'un des deux est en général 0 V. Le schéma de la figure 9.1 représente un* ***signal numérique NRZ*** *– non retour à zéro –. Au cours des pages qui suivent, $T_b$ est le temps bit, D est le* ***débit binaire*** *et vaut :*

$$D = 1/T_b$$

*D est exprimé en bits par seconde ou* **bauds**.

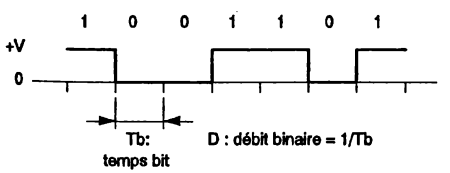



Fig. 9.1
Signal numérique.

## Deux exemples concrets

*Pour comprendre l'importance que revêt la transmission de données numérique, prenons deux cas concrets.*

*En téléphonie, la numérisation du signal audio, préalablement limité à la bande 300-3400 Hz, est effectuée en échantillonnant le signal à la fréquence de 8 kHz puis en codant ces échantillons sur 8 bits. Après numérisation, le signal audio est donc transformé en une source numérique ayant un débit binaire de 64 Kbits/s. En introduisant un codage – compression –, ce débit peut être réduit à 32 Kbits/s sans dégradation de la qualité subjective de la parole. Pour le radiotéléphone cellulaire européen – **GSM** –, ce débit à été réduit à 13 Kbits/s.*

*Pour un signal vidéo à 625 lignes et 25 images par secondes, le signal de* ***luminance*** *Y est échantillonné à 13.5 MHz et les signaux de* ***chrominance*** *U et V à 6.75 MHz. Tous les échantillons sont codés sur 8 bits. De ce codage, il résulte un débit binaire de :*

$D = (13,5) \cdot 8 + 2[(6,75) \cdot 8] + 27 = 243$ *Mbits/s, dont 27 Mbits/s sont réservés à des*

*fonctions annexes. Les techniques de compression permettent actuellement de réduire ce débit à 32 Mbits/s sans altérer la qualité subjective des images. Pour des applications spécifiques, il est encore possible de réduire ce débit. Pour le visiophone par exemple, le signal est réduit à 64 Kbits/s au prix évidemment d'une dégradation notable de la qualité d'image.*
*Dans le cas de la télévision numérique, norme* ***MPEG2****, ce débit est réduit en deça de 10 Mbits/s.*
*Ces deux cas simples mais extrêmement concrets montrent que l'on dispose de* ***sources numériques*** *que l'on cherche à transmettre dans un* ***canal*** *ou milieu de transmission.*

## Milieu de transmission

Le milieu de transmission représente le lien physique entre l'émetteur et le récepteur, il peut être constitué par l'un des quatre éléments suivants :

– un **câble bifilaire** dont la bande passante est faible et qui est en général réservé à des applications à bas débit, inférieures à quelques Mbits/s,

– un **câble coaxial** dont la bande passante est plus élevée que le câble bifilaire et qui autorise des liaisons à quelques centaines de Mbits/s,

– une **fibre optique** ayant une bande passante de quelques GHz qui, associée à une faible atténuation et une bonne immunité aux parasites, la destine aux transmissions intercontinentales à très haut débit,

– l'**espace libre** qui utilise la propagation d'une onde électromagnétique dans l'atmosphère. Ce milieu est généralement réservé aux transmissions par satellites ou par faisceaux hertziens ainsi qu'aux radiocommunications avec les mobiles.

À partir de ces quatre cas, on peut faire une remarque très importante et fondamentale en transmission numérique. Dans les trois premiers cas, le milieu de transmission englobe la fréquence 0 – composante continue. En transmission hertzienne, la composante continue ne peut être transmise et ceci peut poser un véritable problème comme nous le verrons. Lorsqu'il est possible de transmettre la composante continue, c'est le cas des câbles par exemple, on préfère la réserver à la téléalimentation du ou des récepteurs et amplificateurs répéteurs. Il y a cependant des limites en puissance consommée.

## Problème lié au signal NRZ

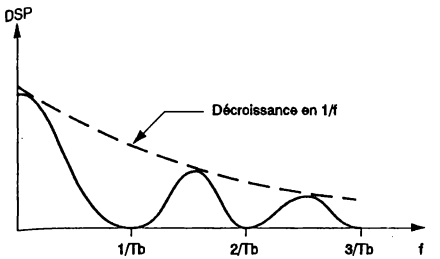



Fig. 9.2
Spectre de densité spectrale de puissance de signal NRZ.

La courbe de **densité spectrale de puissance** (DSP) du signal NRZ de la figure 9.1 est représentée à la figure 9.2. Cette courbe montre que le signal NRZ possède une énergie à la fréquence 0, donc comporte une composante continue. Ceci se comprend de manière intuitive en considérant une suite permanente de 0 ou de 1. La courbe de densité spectrale de puissance est obtenue en analysant un signal numérique aléatoire où les 0 et 1 sont équiprobables.

Grâce au schéma synoptique de la figure 9.3, qui représente un cas concret de transmission, nous allons démontrer que la composante continue pose un réel problème. Le synoptique de la figure 9.3 représente un oscillateur asservi en phase : PLL.

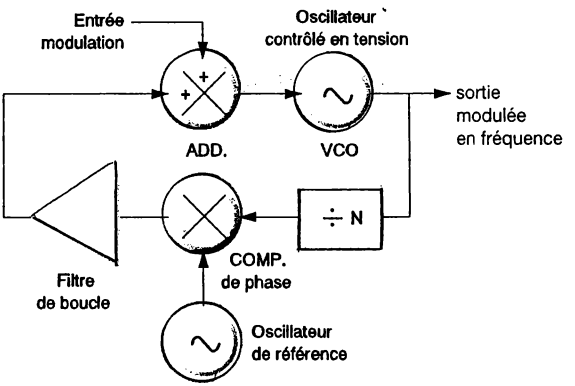



Fig. 9.3
PLL modulé en fréquence.

La figure 9.4 donne la fonction de transfert du système bouclé $G(p)$ et la fonction de transfert pour l'entrée de modulation : $1 - G(p)$. Pour l'entrée de modulation, la fonction de transfert est celle d'un passe-haut. L'entrée de modulation ne peut accepter une composante continue.
Le spectre peut être scindé en deux parties :
- de 0 à $\omega_0$ , action de l'asservissement,
- au delà de $\omega_0$ , signal modulant.

Vouloir moduler un PLL avec un signal comportant une composante continue est un des pièges classiques.

Ceci est électriquement possible et n'a pour effet que de déverrouiller la boucle à accrochage de phase et donc d'annuler et empêcher tout asservissement.

On doit donc faire subir une transformation au signal NRZ pour le rendre apte à passer dans le canal de transmission.

Cette transformation est confiée à des circuits modulateurs-démodulateurs que l'on a pour habitude d'appeler **modems**. Le rôle des modems consiste à transformer le signal NRZ en un autre signal ne comportant pas d'énergie à la fréquence 0.

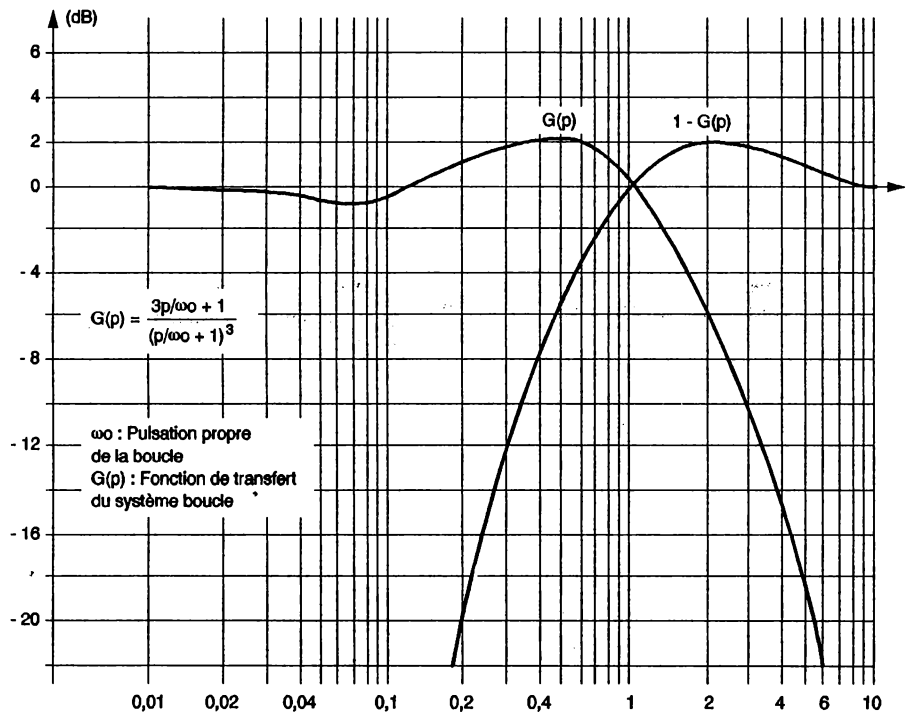



Fig. 9.4

Fonction de transfert du PLL $G(p)$ et pour le signal modulant $1 - G(p)$.

La figure 9.5 résume les deux cas de transmission **asynchrone** ou **synchrone**. Pour la transmission synchrone, on combine les données et l'horloge de cadencement. Pour les transmissions asynchrones, l'horloge peut être reconstituée à partir du signal transmis et l'on parle alors de restitution ou reconstitution d'horloge. Cette opération est en général le problème le plus difficile à résoudre.

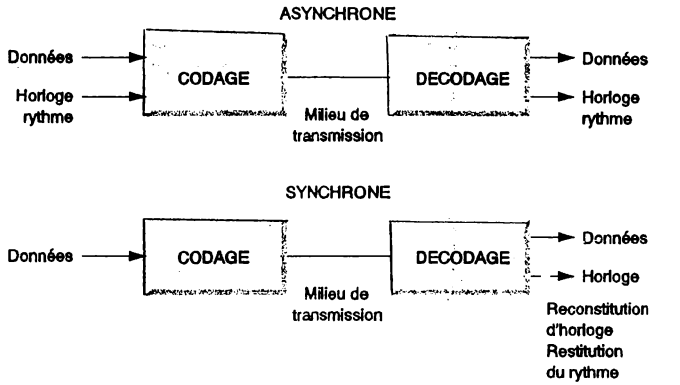



Fig. 9.5
Transmissions asynchrone et synchrone.

Finalement, on distingue les transmissions en bande de base et les transmissions sur onde porteuse. Lorsque la bande *B* allouée à la transmission est comprise entre 0 ou une fréquence proche de 0 et une fréquence $F_{max}$, la transmission est dite en bande de base. Si la bande de fréquence *B* allouée à la transmission est centrée autour d'une fréquence $F_0$, on parle alors de transmission sur onde porteuse. Notons enfin que l'on réserve plutôt le terme de modulation pour les transmissions sur onde porteuse et que l'on parle de codage pour les transmissions en bande de base. Cette introduction aux transmissions numériques montre que le choix du modem ou du procédé de codage résultera d'une optimisation sur les paramètres suivants :
- probabilité d'erreurs,
- efficacité spectrale : rapport entre le débit binaire et la bande occupée,
- complexité et coût.

## Transmission en bande de base

### 1 Probabilité ou taux d'erreur bit

La mesure de la probabilité d'erreur est facilement accessible. Il s'agit en fait du rapport du nombre de bits faux sur le nombre de bits transmis. Par bit faux, on entend détection d'un 1 alors que 0 était transmis ou l'inverse, la détection d'un 0 alors que 1 était transmis. Pour une transmission en bande de base, la probabilité ne dépend pas de la forme d'onde mais uniquement de son énergie. Deux formes d'onde différentes mais ayant une même énergie conduisent donc à la même probabilité d'erreur. La courbe

de la figure 9.6 donne le taux d'erreur bit en fonction du rapport signal sur bruit sur un canal idéal. Cette courbe s'applique aux codes NRZ et biphase.

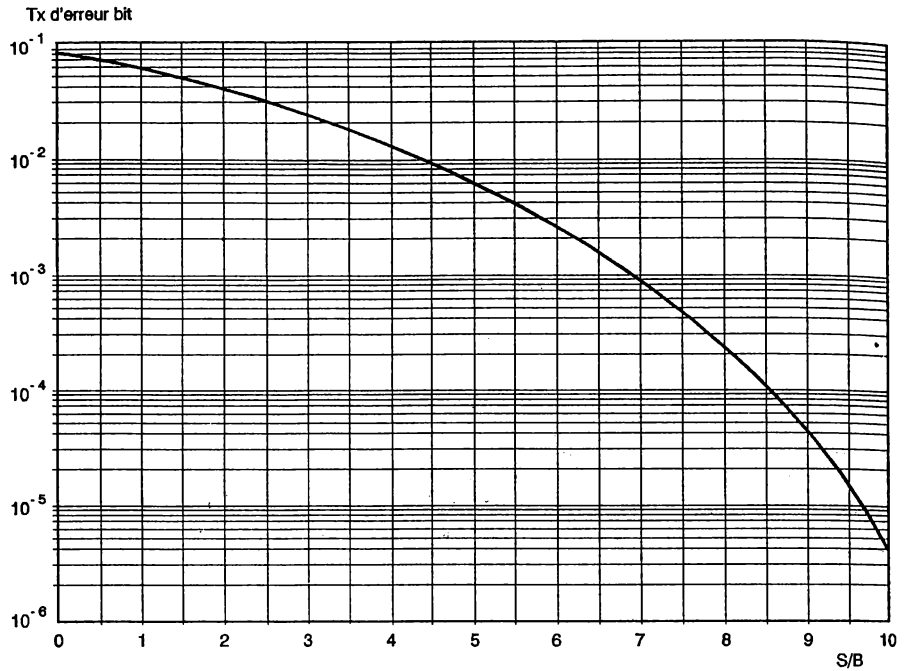



Fig. 9.6
Taux d'erreur bit en fonction de $S/B$ pour un signal en bande de base.

La courbe de la densité spectrale de la figure 9.2 appelle un commentaire important. La DSP du signal est constituée d'arches de largeur $1/T_b$ et la première arche comprise entre 0 et $1/T_b$ contient le maximum d'énergie. En transmission numérique, on démontre qu'il est inutile de conserver les arches d'ordre supérieur à 1, le but à atteindre étant d'optimiser le critère d'efficacité spectrale. Pour le signal NRZ, la bande minimale nécessaire à la transmission vaut : $\frac{1}{2 \cdot T_b}$ (critère de Nyquist). Ceci constitue le premier résultat intéressant. Il est inutile de chercher à transmettre les composantes de fréquence élevée du signal numérique. Dans la pratique, on admet, pour le signal NRZ, une limitation de fréquence légèrement inférieure à $\frac{1}{T_b}$. Pour cette raison et dans la suite de ce chapitre, seule la première arche de la DSP d'un signal est représentée.

La limitation de bande fait obligatoirement appel à des filtres et on suppose que la théorie du filtrage est connue. En transmission numérique, le choix du filtre est important et il conditionne les résultats en terme de taux d'erreur. Cette configuration, filtrage du signal numérique puis transmission des signaux, surprend toujours les électroniciens qui n'ont jamais été confrontés aux problèmes de transmission numérique. Puisque la bande passante est réduite, les temps de montée et descente diminuent et on est en présence non plus d'un signal logique mais plus proche d'un signal analogique. À la réception, un comparateur, que l'on appelle organe de décision dans les traités de transmission numérique, restitue le signal original. De cette caractéristique, il résulte que les récepteurs ou démodulateurs sont toujours plus complexes que les émetteurs ou modulateurs.

## 2 Code biphase

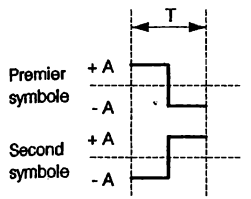



Fig. 9.7
Deux symboles possibles pour le code biphase.

Un autre code est souvent utilisé pour remplacer le code NRZ, il s'agit du code biphase.

Ce code, qui est aussi nommé **Manchester**, est une représentation à deux niveaux, obtenue en transmettant pendant l'intervalle $T_b$ correspondant à un symbole binaire deux polaritées opposées, chacune d'elle occupant un intervalle égal à $T_b/2$. On a donc les deux formes de symbole possible de la figure 9.7.

On peut affecter chacun des deux états logique 0 et 1 à l'un de ces symboles, par exemple 0 au premier et 1 au second. La figure 9.8 donne pour un signal NRZ d'entrée le résultat du codage biphase.

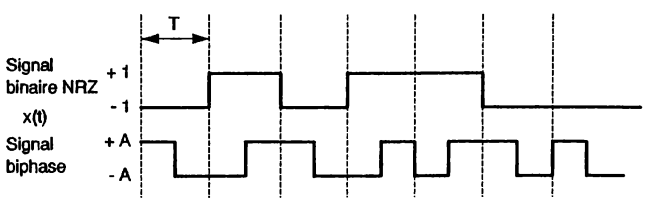



Fig. 9.8
Codage NRZ → codage biphase ou Manchester.

La DSP résultante est donnée à la figure 9.9 qui permet de comparer les DSP des différents codages. Le premier objectif est atteint puisque la composante continue est nulle mais ceci au détriment de l'encombrement spectral.

Dans le cas du biphase, la première arche est comprise entre 0 et $3 \cdot T_b/2$. En pratique, on admet qu'une bonne restitution du signal nécessite une bande comprise entre $T_b/4$ et $3 \cdot T_b/2$.

Le principal intérêt de ce code réside dans la simplicité :
– de mise en œuvre du codeur et décodeur,
– de récupération du rythme de transmission des données puisqu'il existe toujours une transition – front montant ou front descendant – au milieu du temps bit. Cette simplicité se paie en encombrement spectral.

Il existe deux variantes WAL1 et WAL2 du code biphase représentées à la figure 9.9. Les DSP des codes biphase WAL1 et WAL2 sont données à la figure 9.10. À cette figure, on remarque qu'en terme d'efficacité spectrale, le code biphase WAL2 est inférieur au code WAL1. Pour le code WAL2, il y a très peu d'énergie dans le bas du spectre. On peut ainsi séparer le spectre en deux voies, la première voie entre 0 et $T_b/2$ pour transmettre par exemple une voie audio, et une deuxième voie entre $T_b/2$ et $2/T_b$ pour la transmission de données numériques (figure 9.11).

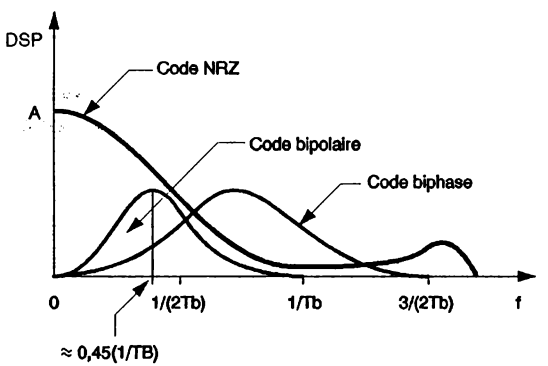



Fig. 9.9
Densité spectrale des codes NRZ, bipolaire et biphase.

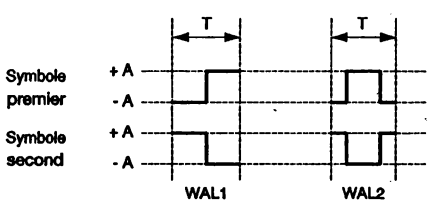



Fig. 9.10
Alternative pour le code biphase.

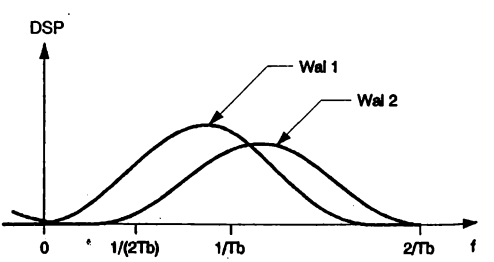



Fig. 9.11
DSP des codes biphase WAL1 et WAL2.

## 3 Code bipolaire

Les codes NRZ et bipolaire sont des codes à deux niveaux. Avec le code bipolaire, on introduit un troisième niveau. La loi de codage est la suivante. Le signal de sortie a une amplitude nulle si le signal NRZ a une amplitude nulle et le signal de sortie prend alternativement les valeurs $-A$ et $+A$ lorsque le signal NRZ est non nul. La figure 9.12 illustre ce principe et la figure 9.13 donne la DSP du signal résultant du codage.

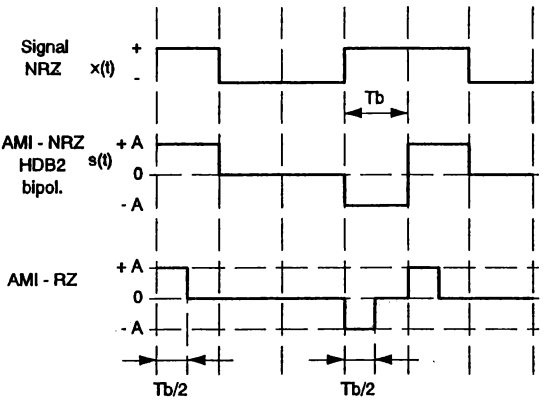



Fig. 9.12
Codage NRZ → bipolaire/HDB2/AMI-NRZ.

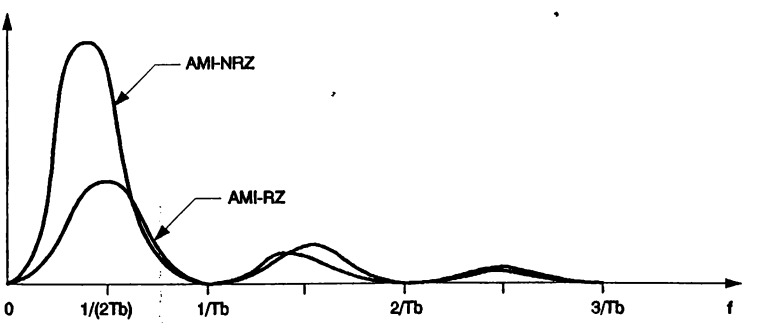



Fig. 9.13
Comparaison des DSP des signaux AMI-RZ et AMI-NRZ.

Pour transmettre ce signal, la bande passante minimale est la même que celle nécessaire dans le cas du NRZ : $1/2T_b$. Avec ce code, il n'y a pas de gain vis-à-vis de l'efficacité spectrale, le seul gain est l'annulation de la composante continue. Ce code porte aussi l'appellation AMI-NRZ. Une de ses variantes est le code

AMI–RZ aux figures 9.12 et 9.13. Le redressement du signal AMI–RZ fournit un signal RZ binaire dont le spectre s'annule aux multiples de $2/T_b$ et présente une raie à $1/T_b$ propice à la synchronisation du récepteur : c'est la récupération du rythme. Ceci démontre l'intérêt des codes bipolaires ou AMI. Remarquons que de longues suites de 0 impossibles à interdire privent les circuits récupérateurs de rythme de toute information. Afin d'éviter ce problème, on a recours à d'autres codes dits à haute densité.

## 4 Code à haute densité HDB3

Le code **HDB3** a été proposé comme variante du code bipolaire pour limiter le nombre de zéros successifs. Il ne diffère du code bipolaire que lorsque le message initial NRZ comprend une longue suite de zéros. Dans ce cas, une séquence spéciale de remplissage est insérée. Pour que les séquences de remplissage puissent être repérées par le récepteur, elles doivent **différer** de la suite normale. La règle de codage de ces séquences est la suivante :

– tous les éléments de la séquence de remplissage sont des zéros à l'exception du premier et du dernier,

– le dernier élément est un viol de bipolarité,

– le premier élément est soit un 0 soit un 1, il est choisi de telle sorte que la polarité du viol soit différente de celle du viol antérieur.

La figure 9.14 illustre ce codage par un exemple. Les circuits logiques ne peuvent proposer que deux niveaux, 0 ou 1.

Un codeur HDB3 délivre par exemple les signaux $A_{OUT}$ et $B_{OUT}$ et le signal HDB3 est obtenu grâce à un amplificateur différentiel externe.

Si la séquence de remplissage est de $n + 1$ bits, on dit que le code HDB est d'ordre $n$. Dans le cas de la figure 9.14, une suite de 4 zéros est remplacée par une séquence spéciale et le code est HDB3. La densité spectrale du code HDB3 diffère assez peu de celle du modèle bipolaire dont elle dérive.

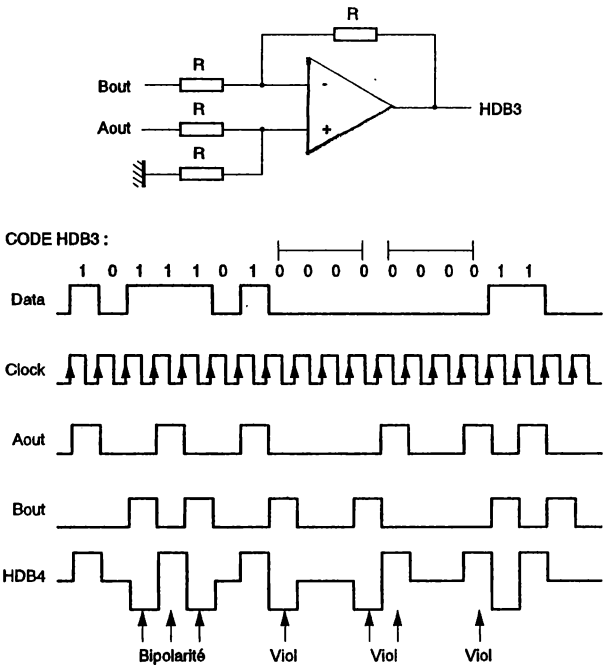



Fig. 9.14
Règle de codage pour le signal HDB3.

## Modulation autour d'une fréquence porteuse

Pour un signal sinusoïdal, l'expression est de la forme :

$$u(t) = A \cdot \sin(\omega t + \varphi)$$

Ce signal peut être modulé en amplitude (action sur $A$), en fréquence (action sur $\omega$) ou en phase (action sur $\varphi$). Bien que les modulations de phase soient très performantes et de plus en plus utilisées, elles sortent du cadre de ce chapitre et sont donc volontairement écartées. Nous traitons uniquement le cas d'une modulation par un signal numérique. Les modulations par un signal analogique sont abordées au chapitre 8 du premier tome.

### 1 Modulation d'amplitude par un signal tout ou rien

Le signal numérique NRZ module directement la porteuse $u(t)$. Le signal résultant a pour expression :

$$u(t) = a_k \cdot A \cdot \sin(\omega t + \varphi)$$

avec : $\omega = 2\,\pi f$.

Dans cette expression, $a_k$ peut prendre les valeurs 0 ou 1. La représentation temporelle du signal est donnée à la figure 9.15.

Puisqu'il s'agit d'une modulation d'amplitude, la DSP du signal binaire NRZ se retrouve de manière symétrique autour de la fréquence porteuse. On peut dire aussi qu'il s'agit d'une translation de fréquence. Comme dans le cas de la transmission en bande de base, le spectre autour de la fréquence porteuse est limité à une valeur comprise entre $f + 1/2{\cdot}T_b$ au minimum et $f + 1/T_b$ au maximum pour répondre au critère d'efficacité spectrale. La limitation peut s'effectuer soit au niveau du signal modulant (limitation de bande du signal NRZ par un filtrage passe-bas) ; soit au niveau de la porteuse par un filtrage passe-bande autour de $f$.

Pour ce procédé de modulation, si le signal NRZ est limité à une fréquence $f_{max} = k/T_b$ , la bande occupée autour de f vaut :

$$B = 2\, f_{max} = 2\, k\, /T_b$$

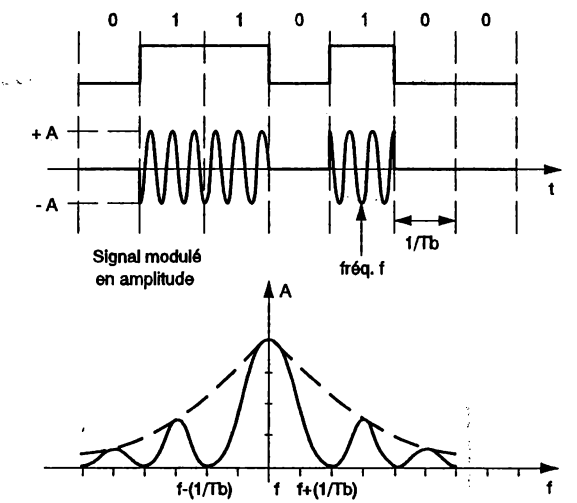



Fig. 9.15

Signal modulé en amplitude, aspect temporel et fréquentiel.

Cette modulation, souvent nommée **OOK**, est mise en œuvre à chaque fois que les critères de complexité et de coût sont primordiaux et que les performances en termes de taux d'erreur et d'efficacité spectrale sont reléguées au second plan. Cette remarque s'applique uniquement aux transmissions hertziennes sans support. Dans les transmissions numériques par fibre optique à haut débit (plusieurs Gbits/s), seul le code OOK est utilisé.

## 2 Modulation par déplacement de fréquence FSK

Pour une première approche, la modulation de fréquence peut se concevoir comme une double modulation OOK obtenue à partir du schéma de la figure 9.16. Au premier symbole binaire, on associe une fréquence $f_1$ et au second une fréquence $f_2$, et l'on pose :

$$f_2 = f + Df$$

$$f_1 = f - Df$$

Le signal de sortie du modulateur ainsi constitué a pour expression :

$$u(t) = A \sin[\omega + (a_k - 1)\, D\, \omega]t$$

où $a_k$ peut prendre les valeurs 0 et 1.

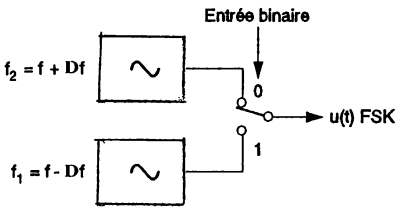



Fig. 9.16
Génération d'un signal FSK à phase discontinue.

Aux instants de commutation, la phase relative des deux générateurs est quelconque. Il en résulte des discontinuités de phase pour le signal $u(t)$. La figure 9.17 rend compte de l'allure du signal de sortie. Le spectre du signal FSK ainsi obtenu (figure 9.18) se compose de l'addition des deux porteuses modulées en amplitude.

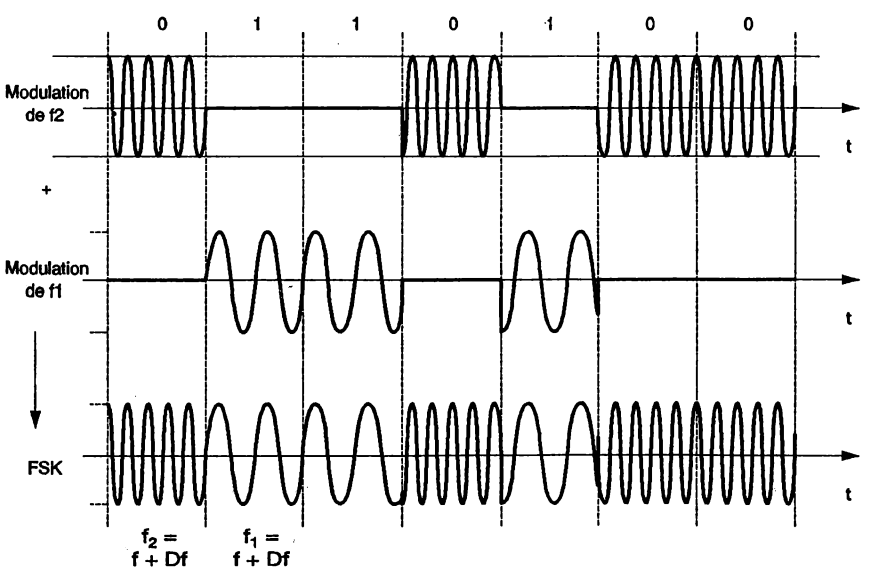



Fig. 9.17
Allure temporelle du signal FSK à phase discontinue.

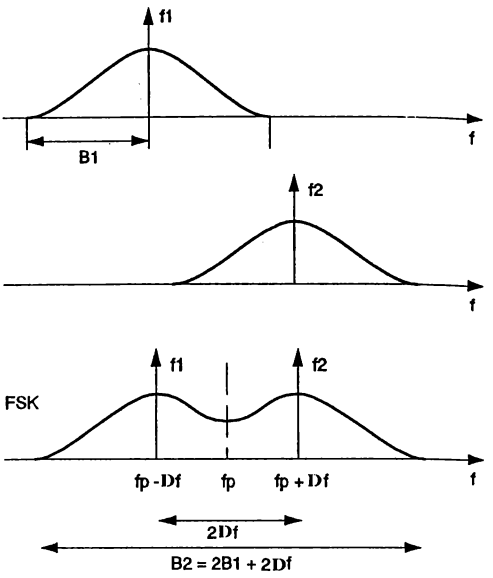



Fig. 9.18
Spectre du signal FSK en phase discontinue.

La bande $B_2$ occupée par le signal FSK est obtenue par la relation :

$$B_2 = 2 \cdot B_1 + 2 \cdot Df$$

où $B_1$ représente la limitation de spectre du signal NRZ. Bien qu'électriquement rien n'empêche la mise en œuvre de ce procédé, il n'est pas utilisé dans la pratique. Cette description est surtout intéressante pour aboutir rapidement à la notion d'encombrement spectral.

Dans la pratique, les deux fréquences $f_1$ et $f_2$ sont, par exemple, issues d'un même oscillateur contrôlé en tension : VCO.

Dans ces conditions, la discontinuité de phase du signal disparaît et la DSP est modifiée. Les raies discrètes aux fréquences $f_1$ et $f_2$ disparaissent, l'enveloppe de la DSP présente des maxima espacés approximativement de $f_2 - f_1 = 2 \cdot Df$ et d'autant plus accentués que $Df$ est grand par rapport à $B_1$, donc au débit binaire $1/T_c$.

Ces résultats sont consignés dans les courbes de la figure 9.19. Pour satisfaire au critère d'encombrement spectral, il est conseillé de choisir des valeurs de x comprises entre 0,5 et 0,75.

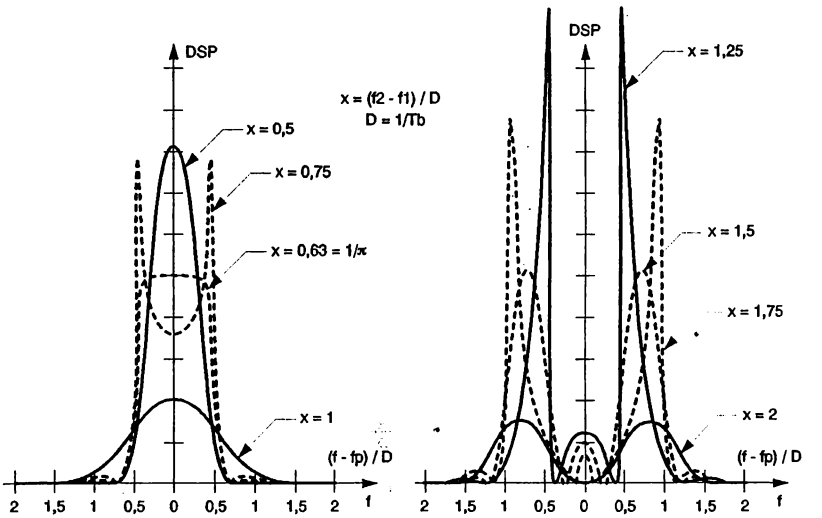



Fig. 9.19
Spectre FSK en fonction de l'écart des porteuses.

Notez qu'il existe un cas particulier *MSK* pour Minimum Shift Keying assez intéressant et traité dans le paragraphe suivant, dans une application à 1 200 bauds.

Dans la pratique, on trouve les valeurs de la figure 9.20.

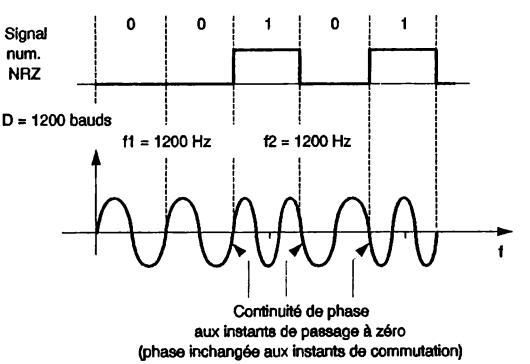

Signal num. NRZ
0 0 1 0 1
D = 1200 bauds
f1 = 1200 Hz
f2 = 1200 Hz
f
Continuité de phase
aux instants de passage à zéro
(phase inchangée aux instants de commutation)

Fig. 9.20
Continuité de phase aux instants de passage à zéro.
Phase inchangée aux instants de commutation.

On a donc :

$$f_2 = f + Df = 1\,800 \text{ Hz}$$

$$f_1 = f - Df = 1\,200 \text{ Hz}$$

$$D = 1\,200 \text{ bauds.}$$

Ces paramètres donnent x = 0,5. Sur les courbes de la figure 9.19, on note que cette valeur donne la meilleure efficacité spectrale. Dans les modems intégrés, les deux porteuses à 1 200 et 1 800 Hz ne sont pas générées par un oscillateur commandé en tension mais reconstituées numériquement par un oscillateur à quartz par exemple à 1 008 MHz. Ceci permet d'assurer numériquement la continuité et l'égalité de phase aux instants de commutation du signal NRZ.

La comparaison entre la modulation AM ou FSK n'est pas aussi simple qu'on pourrait le penser. Pour avoir un taux d'erreur bit de $10^{-5}$, il faut un rapport $S/B$ de 10 dB en AM et 12,2 dB en FSK. Pour effectuer un choix correct, il faut prendre en compte la présence de bruit ou de brouilleurs.

**Conclusion**

*Ce condensé de transmission numérique a pour but de montrer les avantages et inconvénients des principaux procédés de codage et de modulation. Tous les paramètres importants sont ici réunis pour permettre un choix judicieux au moment de la conception et opter pour le meilleur compromis : taux d'erreur, efficacité spectrale, complexité et coût. Il faut retenir qu'il n'existe pas de procédé universel capable de satisfaire les trois critères mentionnés.*

*Dans la plupart des cas, les fonctions de codage et décodage peuvent être confiées soit à un DSP soit à un microcontrôleur. L'importance des télécommunications a cependant permis le développement de circuits intégrés spécifiques pour des codages ou modulations particulières.*

Chapitre 10

# CIRCUITS D'INTERFACE POUR BUS ET RÉSEAUX

## SOMMAIRE

- Liaison série et liaison parallèle, bus et réseaux
- Exemple du bus PC
- Transmission synchrone ou asynchrone
- Les standards de communication
- Les UART

*L'information codée en binaire se présente sous la forme d'une série de mots de n bits. Les valeurs 8, 16 et 32 sont les plus fréquentes et sont reliées aux technologies des microprocesseurs. Pour faire communiquer les circuits intégrés entre eux, deux modes sont possibles, le mode série et le mode parallèle.*
*Dans le mode parallèle, toutes les sorties d'un circuit sont simultanément envoyées sur les entrées de l'autre. La manière la plus simple consiste à les relier par une connexion électrique. L'exemple de la liaison entre un convertisseur et un microprocesseur illustre cette manière de procéder.*

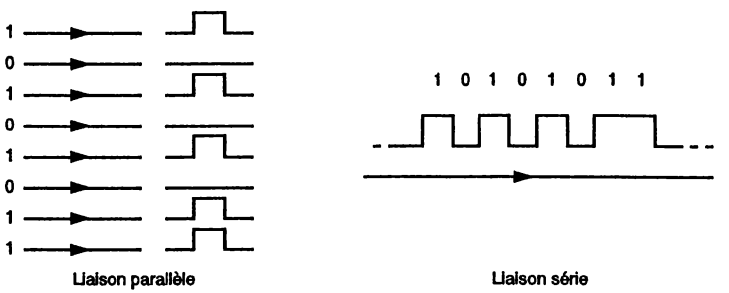



Fig. 10.1
Liaisons série et parallèle.

*Dans le mode série, les bits d'un mot sont transmis successivement les uns après les autres et le procédé est répété pour tous les mots à transmettre. La figure 10.1 illustre les deux modes de transmission et le transfert du mot 10101011 est pris comme exemple.*
*Dans ce chapitre seront étudiés les avantages et les inconvénients des deux modes de transmission ainsi que les principes de fonctionnement et d'utilisation des circuits permettant de gérer le trafic.*
*Le sujet est très vaste puisqu'il introduit tous les standards de communication utilisés en électronique. Aussi nous contenterons-nous d'expliquer les principes généraux de*

© Dunod. La photocopie non autorisée est un délit.

*transmission de données entre circuits et de donner quelques exemples. Le bus des ordinateurs personnels sera détaillé étant donné son usage quasi-universel.*
*Ce chapitre permettra également de définir les termes techniques les plus importants dans la compréhension des « Data books » des constructeurs et de disposer des critères de choix indispensables. Les principaux standards pour les bus et les liaisons numériques seront également présentés.*

# Liaison série et liaison parallèle, bus et réseaux

## 1 Liaison série et liaison parallèle

Dans la transmission des données numériques, quatre niveaux d'interconnexions sont généralement envisagés :
- les communications entre les circuits intégrés d'une même carte ;
- les communications entre cartes dans un même coffret. Un coffret regroupe plusieurs cartes (une dizaine en moyenne) dans une mécanique standardisée. Un ordinateur personnel est construit autour d'un coffret ;
- les communications entre coffrets à courte distance. Par courte distance, on entend des liaisons inférieures à 100 mètres environ. Un ordinateur personnel sera par exemple relié à une imprimante construite elle aussi dans un coffret ;
- les communications à longue distance entre coffrets. Pensons par exemple à un réseau d'ordinateurs personnels. Les distances peuvent alors atteindre plusieurs kilomètres.

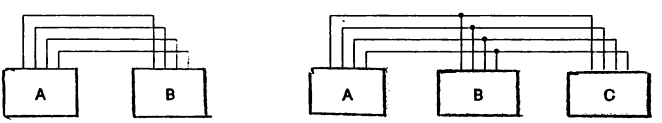



Fig. 10.2
Bus de communication.

Pour faire communiquer deux circuits intégrés sur une même carte, le mode parallèle est très souvent utilisé. La figure 10.2 montre les deux cas classiques : la liaison de deux circuits *A* et *B* et le cas plus complexe où plusieurs circuits sont interconnectés. Dans l'exemple, nous nous sommes limités à trois circuits. Dans ce cas, des lignes électriques sont donc partagées par plusieurs circuits, ce qui introduit la notion de **bus.** Le bus est l'ensemble de ces lignes partagées. Il faut remarquer que, dans le cas simple où un circuit est producteur de données et où les autres sont

consommateurs, la structure du bus ne pose pas de problème particulier. Il faut simplement s'assurer que le producteur de données supporte la charge électrique formée par l'ensemble des consommateurs. Quand les circuits sont à la fois producteurs et consommateurs sur les mêmes lignes électriques, la situation se complique un peu et il faut faire usage de circuits dits à trois états comme il a été vu précédemment.

Un deuxième point important est de noter que la connexion cuivrée visible sur le circuit ou sur le schéma ne constitue que la moitié de la liaison électrique. Il y a toujours un conducteur de retour. Souvent, le plan de masse fait office de conducteur de retour. Le schéma 10.3 établit la correspondance entre la représentation schématique et l'implantation réelle des composants sur la carte. Il ne faudra jamais oublier ce conducteur de retour. Il permet au courant $i(t)$ émis de se refermer. Dans bien des cas, il faut résoudre des problèmes de couplage entre circuits, ce qui génère de fausses informations. La prise en compte de ce conducteur de retour permet d'optimiser le découplage des circuits et la disposition géométrique des composants et des connexions.

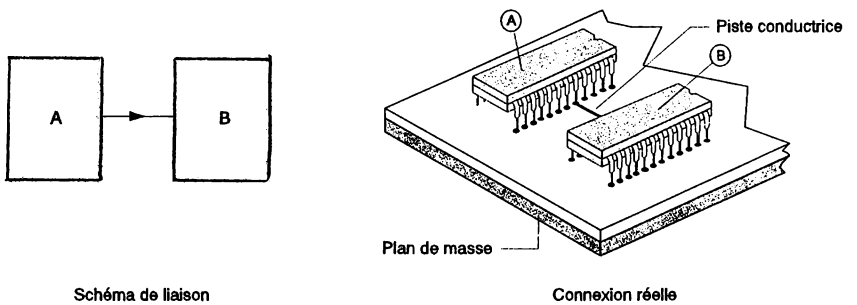
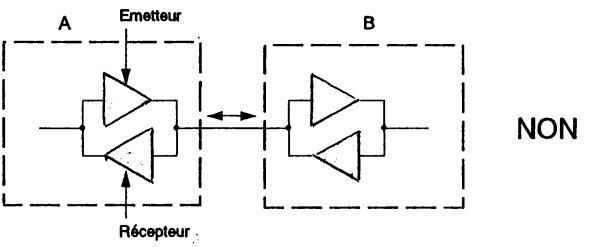


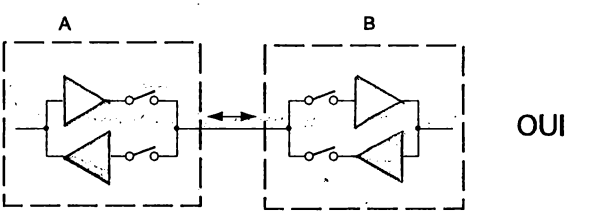

Fig. 10.3
Liaison électrique réelle.

Revenons maintenant sur le cas d'une ligne de connexion entre deux circuits, apte à transporter l'information dans les deux sens : de *A* vers *B* et de *B* vers *A*. Les circuits sont indistinctement émetteur ou récepteur, producteur ou consommateur. La liaison est alors qualifiée de **bidirectionnelle.** La figure 10.4 illustre les difficultés électriques qui apparaissent dans ce type de liaison. Dans le schéma 10.4.a, la sortie du circuit A comprend simplement un circuit émetteur et un circuit récepteur branchés en parallèle sur la sortie logique. La mise en application de ce montage serait catastrophique. Ce montage forme en effet un oscillateur parfait puisqu'il réinjecte en phase le signal de sortie sur l'entrée. Il faut donc imaginer autre chose. Le schéma 10.4.b

a été introduit. Des interrupteurs logiques, en série avec l'émetteur et le récepteur, permettent de sélectionner la fonction convenable (émission ou réception) en fonction de l'état du circuit considéré (émetteur ou récepteur). Ce montage est du type montage à trois états. Le troisième état correspond à l'ouverture simultanée des interrupteurs et assure une isolation électrique quasi-parfaite.




*10.4.a Liaison directe*




*10.4.b Liaison avec circuit type 3 états*

Fig. 10.4
Liaison bidirectionnelle.

Le lecteur comprendra facilement que le mode parallèle est adapté aux liaisons de courte longueur, par exemple pour relier les différents circuits d'une carte. La vitesse de transmission de l'information est alors la plus élevée possible et généralement limitée par la rapidité intrinsèque des circuits. Pour des vitesses très élevées cependant, au-dessus de 100 MHz, l'interconnexion contribue également à la limitation de la vitesse de transfert de l'information. Dans l'avenir, il est envisagé de remplacer les liaisons électriques par des liaisons optiques. Pour relier des cartes électroniques entre elles dans un même coffret, le mode parallèle est généralement utilisé, les longueurs restant compatibles avec la vitesse des circuits. L'interconnexion est alors formée d'un ensemble de lignes électriques parallèles, comme le montre la figure 10.5. Ces lignes sont en nombre plus ou moins élevé et sont le plus souvent le prolongement de la structure interne du microprocesseur utilisé dans le système. On y trouve

donc les lignes de données. Leur nombre est le nombre de bits du microprocesseur : 8, 16 ou 32. On y trouve également les lignes d'adresses (de 8 à 32) et un certain nombre de lignes de contrôle permettant de gérer le trafic sur le bus. Le bus interne des micro-ordinateurs de type PC est conforme à ce schéma général et sera détaillé par la suite.

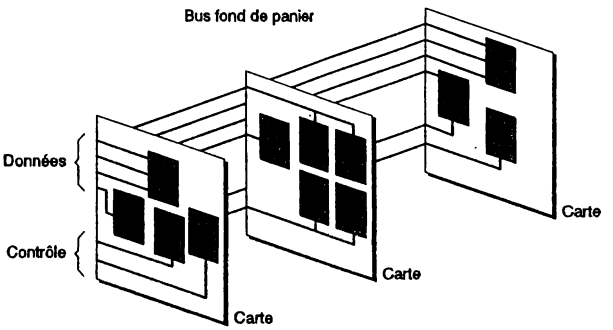



Fig. 10.5
Interconnexion entre cartes.

Quand il faut faire communiquer des coffrets entre eux sur des distances importantes, le mode parallèle devient inadapté. D'une part, l'utilisation de câbles de grande longueur, comprenant un grand nombre de conducteurs, conduit à un coût important. D'autre part, il est difficile d'éviter un couplage entre les diverses connexions quand la longueur est importante. Toutes ces raisons conduisent à privilégier la liaison série représentée en figure 10.6. Les bits sont alors transmis les uns derrière les autres et une ligne électrique unique est utilisée. Elle est, par exemple, formée d'un conducteur et d'une tresse conductrice faisant office de conducteur de retour. Dans certains cas, on utilise simplement deux conducteurs électriques ; la sérialisation des données est réalisée par un circuit particulier : le registre à décalage. Le fonctionnement de ce circuit est détaillé ci-après.

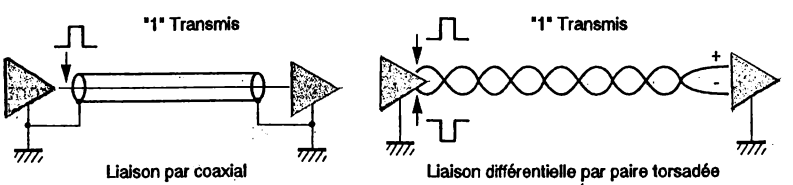



Fig. 10.6
Liaison série.

## 2 Sérialisation et désérialisation des données par un registre à décalage

Dans un premier temps, il est plus simple d'étudier le fonctionnement du circuit réalisant l'opération inverse, c'est-à-dire la transformation d'un message série en un message parallèle. Le schéma est représenté figure 10.7. Le message est supposé organisé en mots de 4 bits. Ces bits sont $B_0$, $B_1$, $B_2$ et $B_3$ et sont appliqués sur les entrées « $D$ » des diverses bascules formant le circuit.

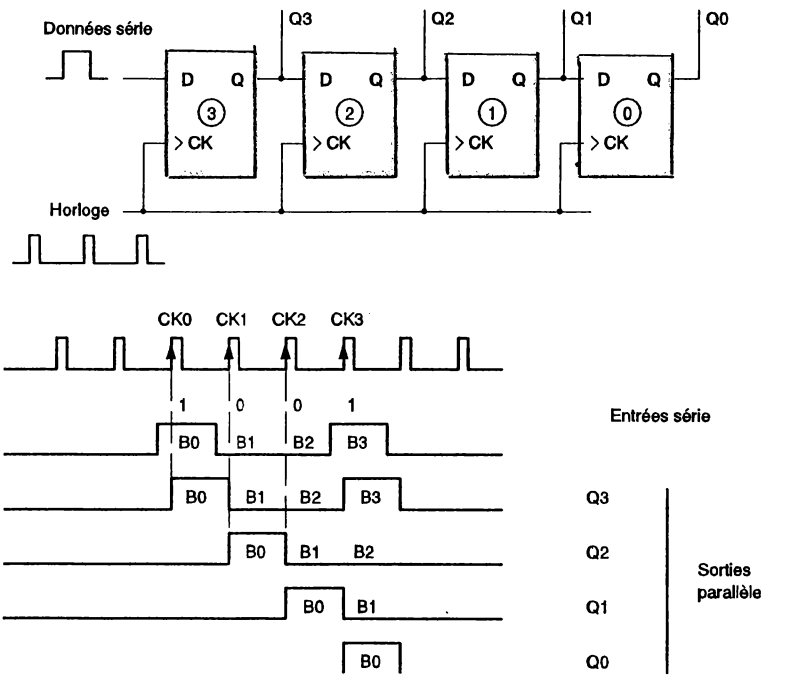



Fig. 10.7
Conversion série-parallèle.

Le fonctionnement de chaque bascule a été décrit en détail au chapitre 2 (logique séquentielle). Il ne sera pas rappelé dans ce paragraphe. À chaque impulsion d'horloge ($CK$), le niveau appliqué en entrée « $D$ » de la bascule est recopié en sortie.

Dans l'exemple donné, le message 1001 est appliqué en entrée. Le bit $B_0$ se présente le premier sur le bistable 3. Sur le front montant de l'impulsion d'horloge $CK_0$, la sortie $Q_3$ passe à l'état haut.

Sur le front montant de la deuxième impulsion d'horloge $CK_1$, la sortie $Q_3$ est recopiée sur la sortie $Q_2$ du bistable 2 et simultanément, le bit $B_1$, appliqué en entrée, est recopié sur la sortie $Q_3$.

Sur le front montant de l'impulsion $CK_2$, la sortie $Q_2$ est recopiée sur la sortie $Q_1$ et simultanément la sortie $Q_3$ est recopiée sur la sortie $Q_2$. Le bit $B_2$ est recopié sur la sortie $Q_3$.

Au dernier coup d'horloge $CK_3$, la sortie $Q_1$ est recopiée sur $Q_0$ et simultanément, $Q_2$ est recopiée sur $Q_1$, $Q_3$ sur $Q_2$ et le bit $B_3$ sur $Q_3$.

En définitive, le bit $B_0$ a été translaté en $Q_0$, le bit $B_1$ en $Q_1$, le bit $B_2$ en $Q_2$ et le bit $B_3$ en $Q_3$. Les différents états électriques sont représentés sur la figure en fonction du temps. On constate donc que l'opération de passage du mode série au mode parallèle a été réalisée.

La transformation inverse, du mode parallèle au mode série, est plus difficile à mettre en œuvre. La figure 10.8 donne un exemple de circuit assurant cette fonction. Le fonctionnement est un peu plus complexe et l'analyse des chronogrammes est proposée au lecteur à titre d'exercice.

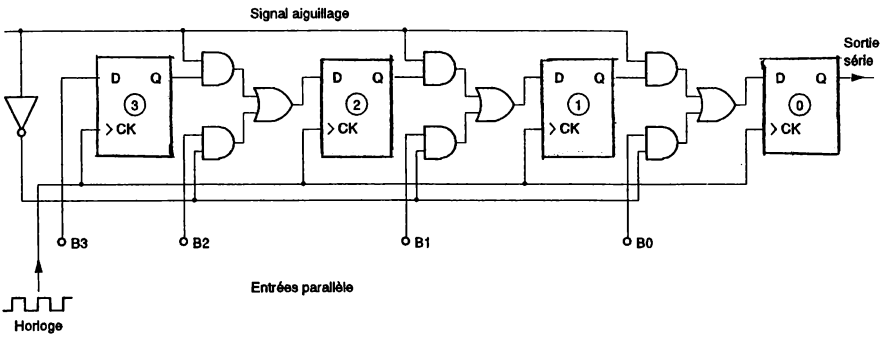



Fig. 10.8
Conversion parallèle-série.

## 3 Bus et réseaux

Comme nous l'avons vu, la transmission série permet donc de faire communiquer facilement deux circuits éloignés. Comment faire quand trois circuit ou plus doivent entrer en communication ? Il y a trois façons de procéder : l'architecture en **étoile**, l'architecture en **marguerite** (daisy chain) ou l'architecture de type **réseau.** La figure 10.9 schématise les trois possibilités.

Dans l'architecture en étoile, un circuit maître M, pourvu de plusieurs sorties ou d'une seule multiplexée, est en liaison avec différents circuits 1, 2 et 3. Les circuits communiquent entre eux par l'intermédiaire du maître.

Dans le système en marguerite, les stations communiquent de proche en proche en réémettant le message reçu.

Dans l'architecture en réseau, les différents circuits sont simplement connectés tous en parallèle sur un support de transmission unique (câble coaxial ou ligne bifilaire). Ces supports ont été décrits au chapitre 8 du premier tome. Cette architecture est d'un usage très courant dans les systèmes électroniques actuels. L'apparente simplicité de ce mode de liaison ne doit cependant pas masquer les difficultés aussi bien de nature électrique que de nature logique.

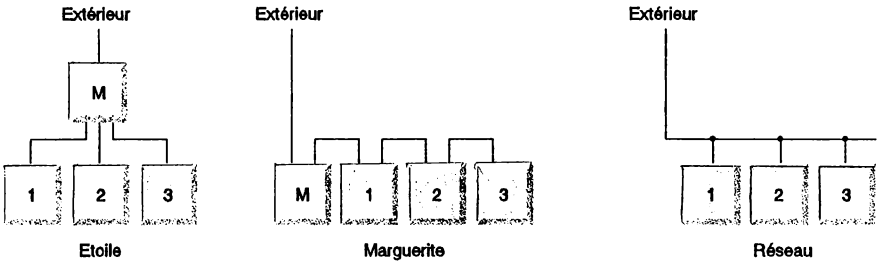



Fig. 10.9
Topologie d'interconnexions.

## Exemple du bus PC

L'usage des PC est si fréquent dans l'électronique actuelle qu'il est indispensable de décrire le bus reliant les diverses cartes constituant le micro-ordinateur. Ces cartes ont été décrites dans le chapitre consacré aux microprocesseurs et microcontrôleurs. Ce sont pour l'essentiel la carte unité centrale, des cartes mémoire et éventuellement des cartes de liaison avec un réseau ou des périphériques. Quand le PC est utilisé dans un système de mesure, on trouve également des cartes d'acquisition de données comprenant des convertisseurs analogique-numérique.

### 1 Le bus PC

La présentation classique du bus est de faire la liste des diverses lignes et des fonctions associées. Une telle présentation est cependant assez fastidieuse dans un ouvrage à but pédagogique. Nous choisirons donc de décrire un certain nombre d'opérations élémentaires de communication entre cartes et nous introduirons les lignes nécessaires au fur et à mesure, quand le besoin sera exprimé. Les lignes évidemment indispensables sont les lignes de données et les lignes d'adresses. Elles sont liées à la nature même du microprocesseur qui associe dans une commande une donnée et une adresse. Les lignes de données sont au nombre de 8, 16, ou 32 selon que le microprocesseur est un 8, 16 ou 32 bits. Les lignes d'adresses sont au nombre de 20 pour le PC/AT et de 32 pour des versions plus récentes.

Étudions dans un premier temps l'**écriture** d'une donnée dans un périphérique. Ce périphérique peut être une imprimante, une carte de communication avec des capteurs ou des actionneurs, une carte d'affichage. Il est nécessaire que le périphérique puisse se reconnaître. Il faut donc lui affecter une adresse (une configuration binaire des 20 bits disponibles). La représentation hexadécimale (définie au chapitre 1 de cet ouvrage) est généralement choisie pour identifier le périphérique. Le déroulement des opérations dans le temps est indiqué figure 10.10.

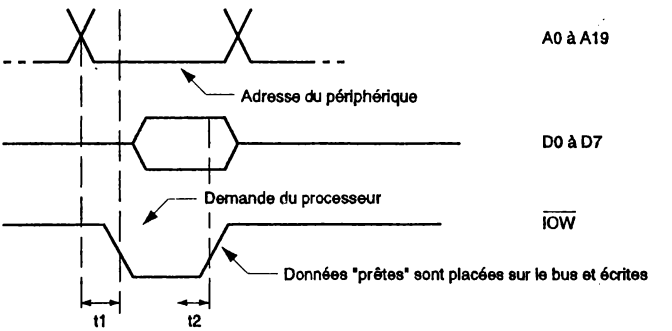



Fig. 10.10
Écriture d'une donnée dans un périphérique.

Après avoir placé les bits d'adresse sur les lignes correspondantes, le microprocesseur doit indiquer que les signaux sont électriquement stables et que les valeurs présentes sur les lignes sont bonnes à l'utilisation. Un signal $\overline{IOW}$ est donc émis sur une ligne particulière, un temps $t_1$ après le positionnement des bits d'adresse. Par convention, ce signal est actif (adresse valide) à l'état bas. Le sigle $\overline{IOW}$ d'origine anglo-saxonne fait référence à « Input Output Write ». La transition descendante de $\overline{IOW}$ sera interprétée par le périphérique comme une demande du processeur.

Le périphérique doit alors se placer en position d'accepter une valeur, par exemple un caractère à afficher. Bien évidemment, le périphérique doit se reconnaître par le décodage des bits d'adresse conformes à son adresse propre. Ce mécanisme a été longuement détaillé dans le chapitre 7. C'est une simple comparaison logique entre la configuration binaire émise sur les lignes de données et celle qui est inscrite dans le périphérique. Le signal $\overline{IOW}$ revient ensuite à l'état haut sur décision du microprocesseur. À cet instant, les données devront être utilisables par le périphérique, ce qui veut dire que le processeur doit avoir positionné les états logiques correspondants sur les

lignes de données, au moins un temps $t_2$ avant le front montant de $\overline{IOW}$. Ce front peut alors être utilisé pour déclencher l'action du périphérique. Cette action est par exemple l'écriture du caractère envoyé en donnée. Elle peut être la conversion analogique des bits de données. Il faut également noter que les données doivent être stables, un certain temps après le front montant de $\overline{IOW}$, afin de garantir un fonctionnement sûr des circuits logiques du périphérique.

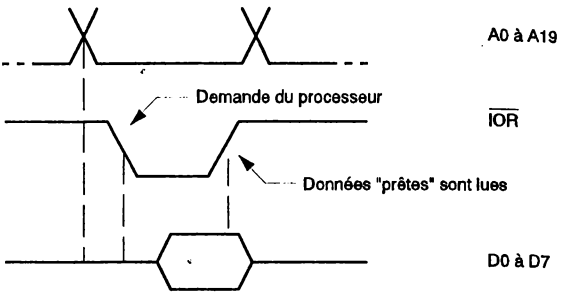



Fig. 10.11
Lecture d'une donnée dans un périphérique.

L'opération inverse, à savoir la **lecture** des données produites par un périphérique, s'effectue selon une procédure similaire. Une autre ligne de contrôle est introduite à cette occasion : la ligne $\overline{IOR}$ (*Input Output Read*). Le schéma 10.11 illustre le déroulement temporel des opérations. De la même manière que pour l'écriture, les bits d'adresse permettant l'identification du périphérique sont placés sur les lignes correspondantes. Sur le front descendant de $\overline{IOR}$, le périphérique ayant décodé son adresse peut alors se mettre en position d'émettre une donnée vers le microprocesseur. La donnée est considérée comme disponible sur le front montant de $\overline{IOR}$. Le périphérique est alors autorisé à placer les états logiques correspondant à la valeur de la donnée sur les lignes correspondantes. Les circuits d'interface avec le périphérique sont nécessairement des circuits à trois états comme il a déjà été expliqué.

Un schéma type d'interface est représenté figure 10.12. On y reconnaît les lignes d'adresse $A_i$ et le circuit de décodage. La sortie du décodeur est mise en ET logique avec le signal de contrôle $\overline{IOR}$ pour donner le signal $\overline{OE}$ (*Output Enable*), prévenant le périphérique qu'il doit préparer les données et autoriser la

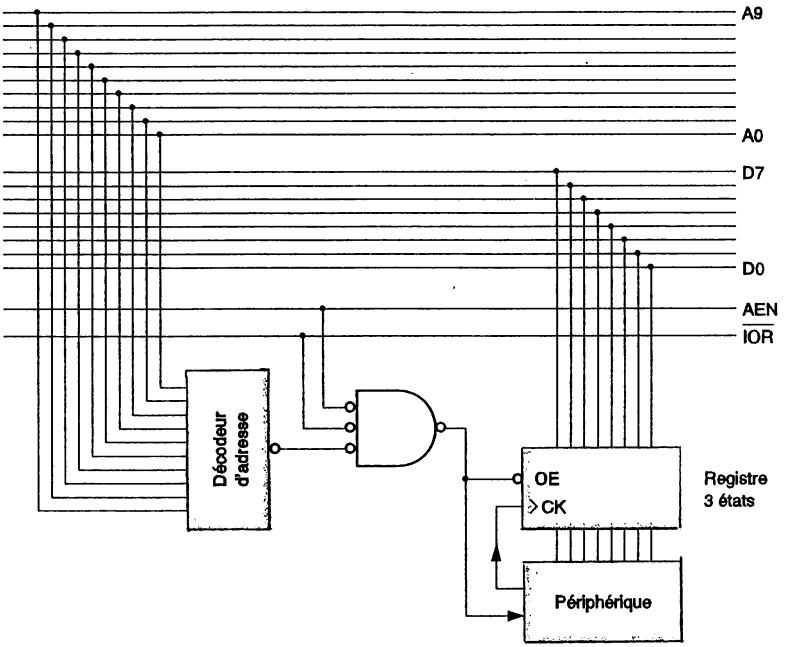



Fig. 10.12
Interface avec le bus.

Un signal de commande ($\overline{AEN}$), non encore décrit, figure sur le schéma. Cette ligne de contrôle permet de sélectionner le mode *DMA* (*Direct Access Memory*). Dans ce mode de fonctionnement, le transfert des données peut alors se faire directement entre le périphérique et la carte d'interface avec le disque dur, ce qui évite de surcharger le microprocesseur.
Avant de décrire une application particulière, il est sans doute utile, à ce stade, de résumer les lignes déjà rencontrées :
- **lignes d'adresses :** émises par le processeur au rythme de son horloge ;
- **lignes de données :** bidirectionnelles et émises par le processeur ou par le périphérique ;
- **ligne IOW :** demande d'écriture du processeur vers le périphérique, émise par le microprocesseur ;
- **ligne IOR :** demande de lecture de données venant du périphérique, émise par le microprocesseur ;
- **ligne AEN :** autorisation de l'accès direct en mémoire, émise par le processeur.

Il est maintenant possible d'illustrer ces diverses fonctions par la description d'une application particulière, en l'occurrence une carte oscilloscope numérique dans le standard PC.

## 2 Description de l'interface d'une carte PC oscilloscope numérique

Pour comprendre le fonctionnement de l'interface, il suffit de savoir que les données, correspondant à l'échantillonnage du signal à visualiser, sont enregistrées dans la mémoire de données de la carte unité centrale. Pour visualiser la trace correspondant à cette impulsion sur l'écran cathodique, il faut alors périodiquement afficher la trace de l'impulsion pendant tout le temps d'observation décidé par l'utilisateur. Si la trace n'était affichée qu'une seule fois, l'utilisateur ne verrait rien puisque la rémanence de l'écran n'est pas suffisante.

La trace de l'impulsion est constituée par un ensemble de points de coordonnées *X* et *Y*. L'abscisse est la position de l'instant d'échantillonnage dans le temps alors que l'ordonnée est la valeur de l'échantillon.

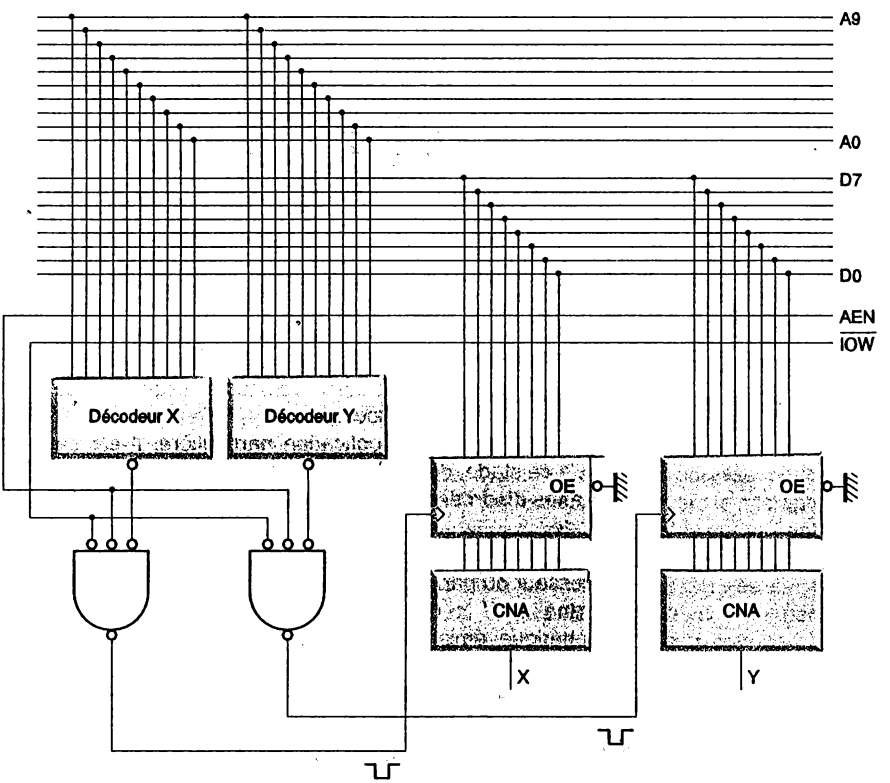



Fig. 10.13
Oscilloscope numérique, interface avec un bus PC.

Le programme de visualisation aura donc à faire évoluer régulièrement la valeur *X* et à présenter les valeurs de *Y* correspondantes. Ces valeurs sont les tensions de déflexion des plaques commandant la position du spot. L'opération se reproduira cycliquement pendant la durée de l'observation. Il est donc nécessaire de disposer de deux registres *X* et *Y*, qui sont considérés comme deux périphériques distincts, donc affectés de deux adresses. L'opération d'affichage est une opération d'écriture dans un périphérique.

Le schéma des circuits d'interface entre le bus du PC et la carte oscilloscope est représenté figure 10.13. On reconnaît dans cette carte les circuits de décodage d'adresse et les convertisseurs numérique–analogique permettant de créer les tensions de déflexion. Les adresses affectées aux registres *X* et *Y* sont 3C0 en hexadécimal pour *X* et 3C1 pour *Y*, soit 11 1100 0000 et 11 1100 0001 en binaire. Les données sont supposées réduites à 8 lignes. Les données sont prises en compte sur le front montant de $\overline{IOW}$. Le programme d'affichage, écrit en assembleur, est donné comme exemple de liaison entre le monde matériel et le monde logiciel. Il est composé de deux modules, un module d'initialisation INI et un module de tracé TRAC.

INI : – initialisation du pointeur *X*
– initialisation du pointeur *Y*
– initialisation du nombre de points à tracer

TRAC : – valeur de *X* placée dans l'ALU
– contenu de la variable *X*, envoyé à adresse 3C0
– valeur de *Y* dans l'ALU
– contenu de la variable *Y*, envoyé à adresse 3C1
– passage à la valeur suivante de *X*
– passage à la valeur suivante de *Y*
– décrémentation du compteur *n* point
– test de *n* point et retour au début de TRAC si *n* point non nul

## 3 Le traitement des interruptions

Tous les cas étudiés jusqu'à maintenant étaient relativement simples dans la mesure où toutes les opérations étaient sous le contrôle du microprocesseur de la carte mère. Que se passe-t-il quand des événements inattendus viennent perturber la régularité des cycles de ce type ? L'étude de cette situation nous permettra de comprendre l'intérêt de nouvelles lignes de

contrôle : **IRQ, DRQ** et **DACK**. L'exemple de la frappe d'un caractère au clavier illustre le cas d'un événement interrompant le déroulement normal d'un programme. Pour gérer cette situation, trois méthodes sont possibles :
- utiliser un registre d'état,
- utiliser les lignes d'interruption du bus,
- accéder directement à la mémoire.

### *3.1 Utilisation d'un registre d'état*

Le principe général est de mettre en mémoire, dans un registre, la demande du périphérique (le clavier par exemple), puis d'inspecter périodiquement l'état de ce registre par le programme principal. Quand l'état « occupé » du registre est lu, il est alors nécessaire de lire le périphérique. Quand le périphérique est lu, le registre est remis à zéro et le programme continue.

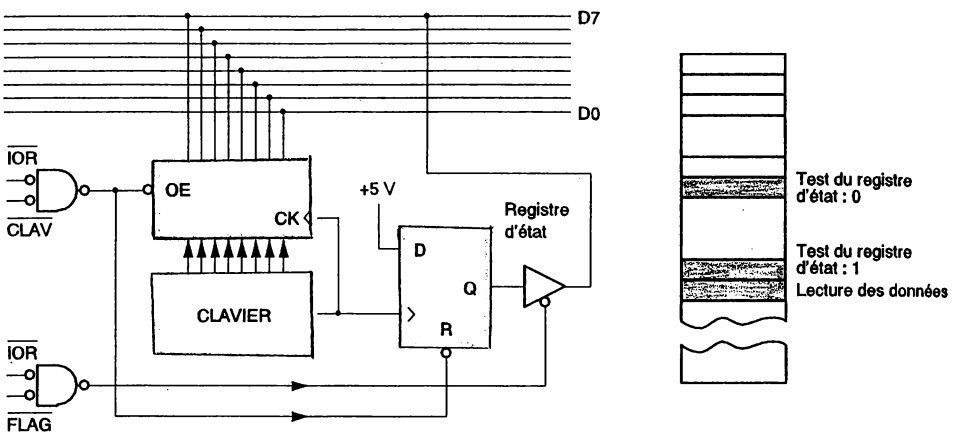



Fig. 10.14
Registre d'état et lecture des données du clavier.

La figure 10.14 montre simultanément les aspects matériels et logiciels de ce mode de fonctionnement dans l'exemple d'une interruption par le clavier. Le clavier a deux adresses, l'une pour lire le caractère tapé et l'autre pour lire l'état du registre d'état. Les décodages d'adresse créent respectivement les niveaux logiques $\overline{CLAV}$ et $\overline{FLAG}$. Les deux lectures possibles sont conditionnées par le niveau $\overline{IOR}$. Le programme teste dans un premier temps l'état du registre, en adressant le registre $\overline{FLAG}$. La lecture se fait alors sur la seule ligne $D_7$ puisqu'un seul bit suffit pour indiquer l'état. S'il est à l'état haut, le programme commande la lecture en adressant le registre du clavier $\overline{CLAV}$. Les huit lignes

de données sont alors utilisées. Le caractère est donc lu puis affiché à l'écran. Le programme se poursuit ensuite après avoir remis à zéro le registre d'état sur l'ordre de lecture du registre du clavier. Dans la pratique, le registre d'état du périphérique n'est pas constitué d'un simple bit mais d'un groupe de bits formant un **mot d'état.** La demande du périphérique y figure mais aussi d'autres indications sur l'état du périphérique : la présence de papier pour une imprimante, la détection d'une erreur de transmission, etc.

### *3.2 Utilisation des interruptions*

L'inconvénient du mode de fonctionnement précédent est dans la nécessité d'interroger fréquemment le périphérique. Quand le périphérique est très actif, par exemple dans le cas d'un disque dur, un autre mode de fonctionnement est préférable : le fonctionnement par interruption. Dans ce cas, le périphérique a un rôle plus actif puisque c'est lui qui interpelle le processeur et lui demande une modification du déroulement du programme.

La figure 10.15 reprend l'exemple d'une interruption demandée par le clavier et présente les aspects matériels et logiciels de la solution retenue.

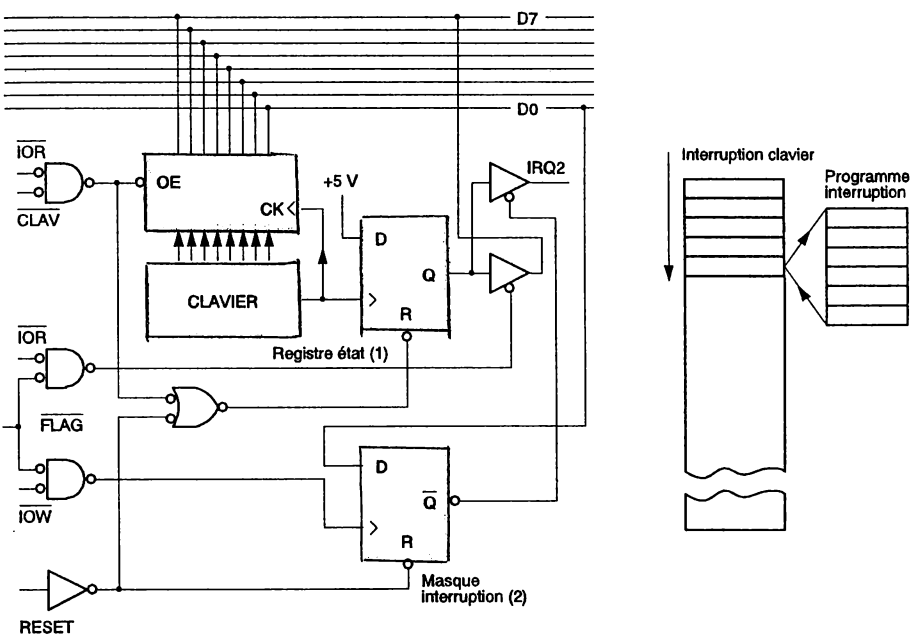



Fig. 10.15

Interruption et programme de lecture des données du clavier.

Les mêmes éléments que ceux de la figure 10.14 sont repris avec en plus l'introduction d'une ligne supplémentaire, la ligne d'interruption $IRQ_2$. En fait, le bus comprend plusieurs lignes numérotées de 0 à 5. Une autre ligne est introduite, c'est la ligne **RESET,** qui transporte un niveau en provenance du processeur. Elle est utilisée comme signal de remise à zéro. Quand le clavier est activé et quand le code du caractère est prêt, un niveau $IRQ_2$ est généré. Ce niveau transite par le registre 1. Le programme principal est alors interrompu et un sous-programme particulier (*Interrupt handler*) dédié au traitement de l'interruption du périphérique est mis en route. Il faut alors sauver tout ce qui doit être sauvé quand le programme principal sera à nouveau en action, par exemple le contenu de tous les registres du microprocesseur.

Le sous-programme, après avoir testé le contenu du registre de statut du périphérique (donnée $D_7$), peut ensuite procéder à la lecture de la donnée correspondant au caractère tapé au clavier dans le registre de sortie du périphérique. Le caractère sera ensuite affiché. Le signal d'interruption peut être rendu inefficace en mettant à zéro le registre 2. Cette fonction est assez utile en pratique. On dit alors que l'interruption est **masquée**. Dans l'exemple proposé, le bit $D_0$ est utilisé à l'adresse *FLAG*. La ligne *RESET* permet une remise à zéro générale du système.
Comme il a déjà été évoqué, le bus PC/XT comporte huit lignes d'interruptions numérotées de 0 à 7. Ces lignes sont respectivement :

- $IRQ_0$ : l'horloge du PC,
- $IRQ_1$ : le clavier,
- $IRQ_2$ et $IRQ_3$ : lignes libres,
- $IRQ_4$ : le port série,
- $IRQ_5$ : le disque dur,
- $IRQ_6$ : le lecteur de disquette,
- $IRQ_7$ : l'imprimante.

Les interruptions sont classées par ordre de priorités décroissantes. En cas de conflit, c'est-à-dire quand deux interruptions ou plus sont émises simultanément, c'est toujours l'interruption du numéro le plus faible qui est traitée la première.

### *3.3 Accès direct en mémoire*

Dans certains cas, les données doivent être transmises rapidement d'un périphérique vers un autre, par exemple une mémoire. Passer par le processeur de la carte mère est possible mais ce n'est pas la solution la plus rapide. Si, par exemple, on utilise une carte d'échantillonnage et si on désire inscrire le contenu des échantillons sur le disque dur, il est nécessaire que le transfert soit rapide étant donné le volume important d'informations à transmettre quand la fréquence d'échantillonnage est élevée.

Le périphérique demande alors l'accès au bus en utilisant les lignes de demande **DRQ**.

Dans le cas du bus PC, trois lignes spéciales sont réservées à cet usage et permettent de traiter trois demandes. Le contrôleur de DMA, circuit complexe spécialisé, prend alors la direction des opérations à la place du microprocesseur. Il signale son intervention en activant une ligne **DACK.** Quatre lignes sont disponibles. Les données sont alors placées sur les lignes de données et les adresses sur les lignes d'adresse, mais les adresses sont celles des emplacements mémoire et non pas les adresses des périphériques. Le transfert peut s'effectuer du périphérique vers une mémoire (le disque dur par exemple) mais aussi de la mémoire vers un périphérique. Pendant cette opération, la ligne *AEN* est activée à l'état haut. Le transfert s'effectue alors par « byte », ensemble de 8, 16 ou 32 bits. Il peut aussi s'effectuer par blocs (succession de bytes). Entre chaque transfert, le microprocesseur peut reprendre la main. Le tableau ci-dessous résume les principales différences entre le mode DMA et le mode normal de fonctionnement.

| | AEN | $A_0$ à $A_{19}$ | Maître du trafic |
|---|---|---|---|
| **mode normal** | bas | périphérique | microprocesseur |
| **mode DMA** | haut | mémoire | contrôleur de DMA |

## 4 Tableau résumé des lignes du bus PC/AT

La figure 10.16 montre les liaisons, via le bus PC, de la carte mère vers les cartes du PC.

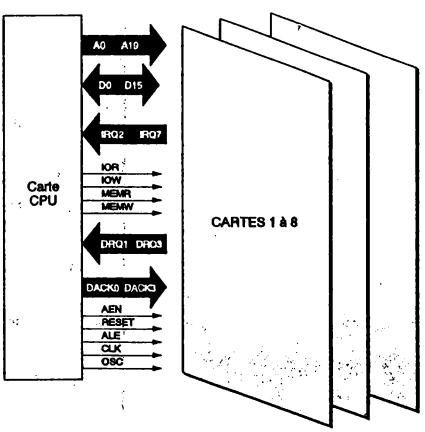



Fig. 10.16
Le bus PC.

Les lignes principales sont caractérisées dans le tableau ci-dessous :

| NOM LIGNE | FONCTION | NATURE | CIRCUIT | ACTIVE | Commentaire |
|---|---|---|---|---|---|
| $A_0$ à $A_{19}$ | lignes adresse | unidirectionnelle | Totem pole | haut | |
| $D_0$ à $D_7$ ou $D_0$ à $D_{15}$ | lignes de données | bidirectionnelle | Trois états | haut | |
| $IRQ_2$ à $IRQ_7$ | interruptions | unidirectionnelle | Totem pole | transition montant | |
| IOR | lecture | unidirectionnelle | Totem pole | bas | |
| IOW | écriture | unidirectionnelle | Totem pole | bas | |
| MEMR | lecture mémoire | unidirectionnelle | Totem pole | bas | |
| MEMW | écriture mémoire | unidirectionnelle | Totem pole | bas | |
| $DRQ_1$ à $DRQ_3$ | interruptions DMA | unidirectionnelle | Totem pole | haut | mode DMA |
| $DACK_0$ à $DACK_3$ | accord de DMA | unidirectionnelle | Totem pole | bas | mode DMA |
| AEN | sélection DMA | unidirectionnelle | Totem pole | haut | mode DMA |
| RESET | initialisation | unidirectionnelle | Totem pole | haut | remise à zéro |
| ALE | adress latch enable | unidirectionnelle | Totem pole | haut | utilisée si multiplexeur |
| CLK | horloge | unidirectionnelle | Totem pole | | |
| OSC | oscillateur | unidirectionnelle | Totem pole | | |

## Transmission synchrone ou asynchrone

Nous avons jusqu'à maintenant étudié les cas simples de transmission de données dans lesquels tous les échanges sont synchronisés par l'horloge du microprocesseur distribuée aux diverses unités du système. Le lecteur doit comprendre que la seule manière de transformer le signal électrique reçu en un message formé d'une suite d'états hauts et bas, est de mesurer le signal en des instants donnés, fixés par les transitions du signal horloge, et de déterminer quels sont les états logiques à ces instants. On effectue finalement une opération d'échantillonnage au rythme de l'horloge.

Quand la distance de transmission augmente, le mode série est généralement choisi pour transmettre les données. Une solution simple consisterait à transmettre simultanément sur une autre liaison le signal d'horloge. Cette solution n'est cependant pas retenue quand les distances sont élevées car elle implique une liaison supplémentaire. De plus, l'horloge est le signal le plus

rapide sur de grandes distances sans le dégrader (effet d'intégration du câble). Ces deux raisons ont donc conduit dans de nombreux cas à ne pas transmettre le signal d'horloge séparément.

Une autre solution vient à l'esprit : c'est d'utiliser plusieurs horloges séparées, une par unité. Cette solution implique cependant des horloges de périodes strictement identiques comme le montre la figure 10.17 qui décrit la communication entre deux unités éloignées. Un message 10010 est émis par l'unité 1 vers l'unité 2. L'horloge 1 a servi à placer les états électriques sur le bus en fonction de données mises en mémoire dans des registres. Les données sont transmises et à la réception, il est nécessaire d'échantillonner le signal reçu pour retrouver les états logiques émis. L'exemple de la figure 10.17 montre qu'une légère différence des périodes des horloges est suffisante pour fausser la transmission. Messages émis et reçu sont différents.

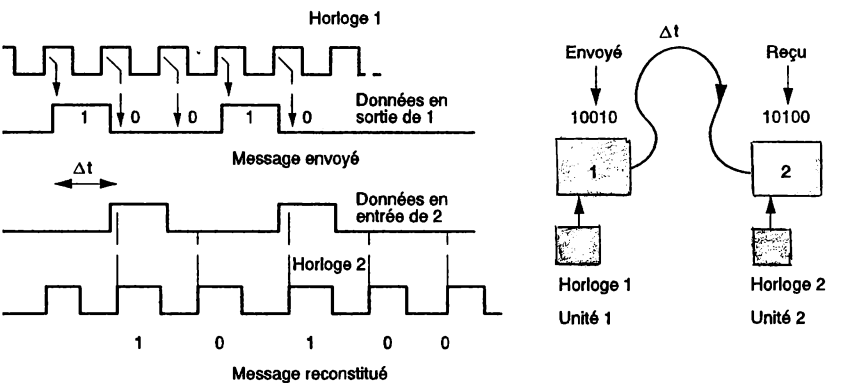



Fig. 10.17
Transmission des données et synchronisation des horloges.

On pourrait également détecter les changements de niveaux des signaux reçus et effectuer la mesure du niveau logique un certain temps après la détection du front en attendant que le signal soit stable. Cette méthode est effectivement utilisée de manière indirecte par les techniques de restitution d'horloge. Certains procédés permettent d'extraire du signal reçu le signal d'horloge qui a servi, à l'émission, à créer les changements de niveaux. Il suffit alors de prendre ce signal comme signal de synchronisation des données reçues comme l'indique la figure 10.17. L'horloge n'est plus créée à la réception *ex nihilo*, mais à partir des transitions du signal reçu. Elle est donc une copie conforme de l'horloge utilisée à l'émission, décalée du temps de propagation dans le câble. Ce procédé est le procédé de base des liaisons

Il est possible d'utiliser une autre technique qui constitue le principe de base des liaisons de type **asynchrone.** Elle consiste à découper l'information en mots de plusieurs bits, chaque mot commençant par un bit de début sans signification particulière et finissant par un bit de fin également non porteur d'information. Ces deux bits ne servent qu'à la synchronisation des données. Le récepteur détecte alors la transition correspondant au bit de début de mot et génère une horloge locale pendant la seule durée du mot transmis. Même si la période de l'horloge ainsi reconstituée n'est pas strictement identique à celle de l'horloge d'émission, aucune erreur ne sera introduite puisque la durée de vie de cette horloge est courte et égale à la durée du mot transmis. Le décalage est donc négligeable et les données sont correctement échantillonnées. À la fin du mot, le bit de fin permet d'interrompre l'horloge du récepteur et place la ligne dans l'état de repos permettant d'accepter un nouveau message.

À la différence de la liaison synchrone assurant un débit continu de l'information uniquement interrompu par des mots non porteurs d'informations et destinés à la reconstitution de l'horloge, la liaison asynchrone fournit un trafic discontinu et irrégulier. Les différents mots sont émis au fur et à mesure que les autorisations d'émettre sont données, généralement après un accusé de réception du module destinataire. Les liaisons asynchrones sont souvent réservées aux communications entre l'homme et la machine (entre un clavier et un processeur par exemple) alors que les liaisons synchrones sont généralement réservées aux liaisons entre machines (liaisons entre un processeur et un disque dur par exemple). Elles permettent également des débits élevés entre processeurs. Les deux types de liaison sont schématisés figure 10.18.

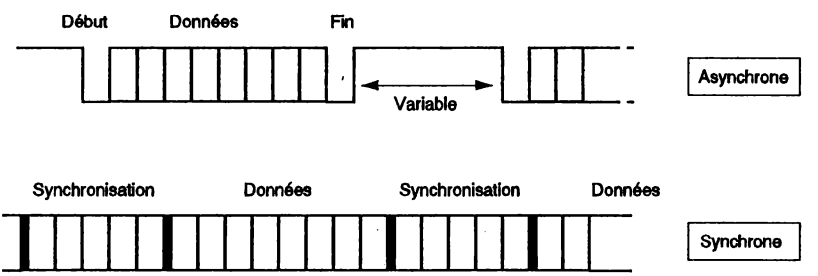



Fig. 10.18
Transmission synchrone et asynchrone.

Pour terminer ce paragraphe difficile mais important, puisqu'il introduit toutes les liaisons rencontrées dans les systèmes électroniques, il est sans doute utile de revenir à des consi-

dérations plus pratiques, par exemple le bus PC qui nous a déjà servi d'exemple d'application. La transmission de type synchrone avec transmission physique de l'horloge a déjà été étudiée dans les exemples précédents.

Étudions maintenant une liaison de type « asynchrone » entre le processeur et un périphérique éloigné. Le cycle d'échanges est généralement le suivant :

- le processeur, dans un premier temps, positionne les bits d'adresse sur le bus ainsi qu'un niveau d'autorisation *IOR* signifiant qu'il désire lire les données du périphérique,
- le périphérique adressé se reconnaît et positionne ses données sur les lignes du bus ainsi qu'un niveau *DTACK* signifiant à la fois que les données sont valides et que la demande a été comprise,
- le processeur voit le niveau *DTACK* et accepte les données émises par le périphérique. Il remet alors le niveau *IOR* à zéro pour pouvoir éventuellement émettre une autre demande.

Il y a eu échange entre le processeur et le périphérique de type « ***handshake*** », de la même manière que deux personnes se saluent par une poignée de mains. Le processeur a fait une demande. Le périphérique a répondu positivement en émettant *DTACK* et le processeur a acquiescé en relâchant le niveau *IOR*. Ce type de transmission, à la différence des échanges étudiés précédemment sur le bus PC, a permis de s'assurer de la présence du périphérique et de son fonctionnement.

Peut-on, dans ce cas, parler réellement de transmission asynchrone au sens où nous l'avons défini dans l'introduction ? La réponse est non, si nous voulons être rigoureux. Dans ce cas simple, aucune horloge n'a été reconstituée au niveau du périphérique et tous les signaux sont en fait synchronisés par l'horloge du microprocesseur. De fait, les échanges sont fondamentalement de type synchrone dans les liaisons par le bus d'un système monoprocesseur. Dans les systèmes multiprocesseurs, le problème de la synchronisation des horloges est à nouveau posé.

## Les standards de communication

Le paragraphe précédent a introduit les bus et les réseaux et a présenté les deux grandes familles de liaisons, les liaisons synchrones et asynchrones. À partir de ces considérations générales, il est maintenant possible de survoler les principaux standards de communication rencontrés dans l'industrie électronique. La notion de standard est très importante. Le lecteur comprendra facilement que pour homogénéiser les équipements et faciliter l'interchangeabilité des matériels, il est indispensable de définir précisément les protocoles de communication et les caractéris-

tiques électriques des liaisons. Seront étudiés successivement : les bus, les liaisons série et parallèle et enfin les réseaux. La figure 10.19 replace les différents termes employés dans le contexte général d'un ensemble de machines de calcul ou de mesure interconnectées par un réseau.

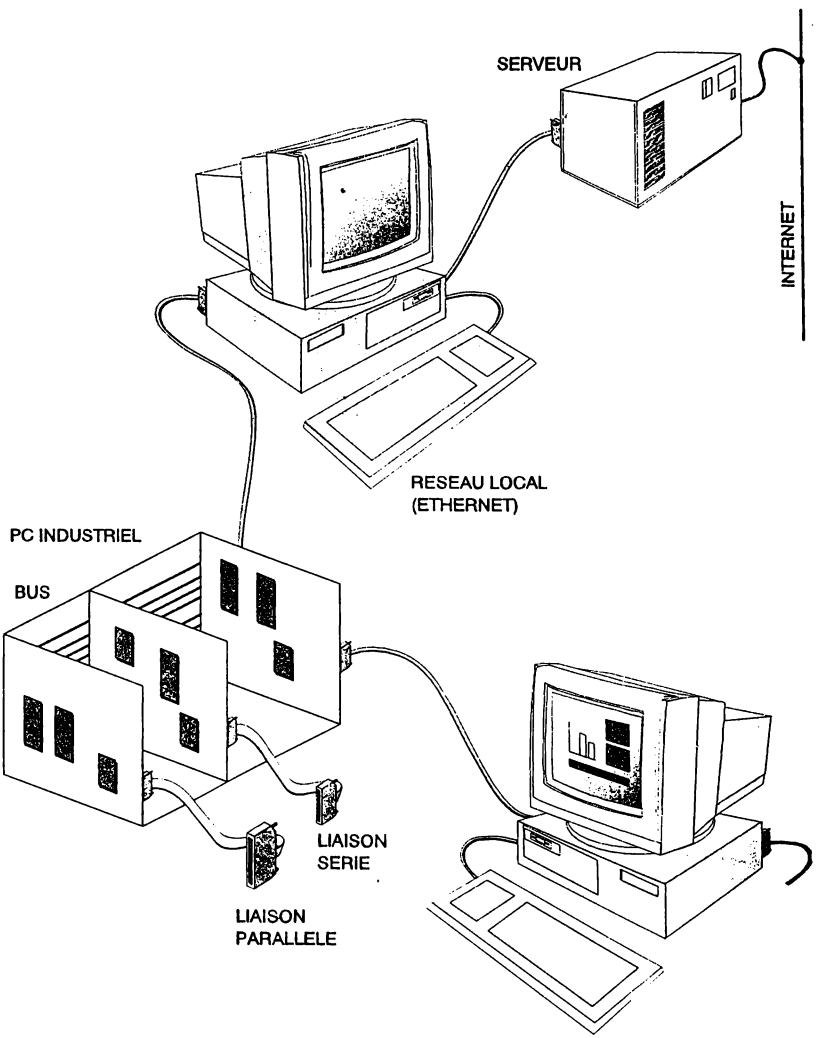



Fig. 10.19

## 1 Les bus fond de panier

Il s'agit d'interconnecter des cartes électroniques sur des distances généralement inférieures au mètre. Le bus des micro-ordinateurs de type PC a été pris comme exemple mais il n'est pas le seul. Tous ces bus ont en commun le fait de présenter des lignes de données, des lignes d'adresses et des lignes de contrôle. Les standards les plus courants sont les suivants :

- PC/XT ou PC/AT, bus EISA : bus pour ordinateur personnel,
- Multibus I ou Multibus II : bus pour station de travail,
- VME : bus pour automatismes industriels,
- NUBUS : bus pour ordinateur Macintosh,
- FUTUREBUS : évolution du bus VME.

Les caractéristiques suivantes seront données :

- la vitesse typique de l'horloge du processeur maître,
- le nombre de lignes de données,
- le nombre de lignes d'adresses,
- le caractère de la liaison : synchrone ou asynchrone,
- la possibilité d'avoir plusieurs maîtres.

Les données et les adresses peuvent se partager dans certains bus les mêmes lignes physiques. Le multiplexage est alors nécessaire. Dans quelques cas, les lignes d'interruption n'existent pas et les interruptions se font en envoyant un message particulier. Le tableau 10.20 ci-dessous résume les principales caractéristiques des bus industriels.

| | PC/XT | PC/AT | EISA | MULTI-BUS I | MULTI-BUS II | VME | NUBUS | FUTURE BUS |
|---|---|---|---|---|---|---|---|---|
| horloge | 1,2 MHz | 5,3 MHz | 33 Mhz et plus | 10 MHz | 40 MHz | 40 MHz | 40 MHz | 120 MHz |
| données | 8 | 8, 16 | 8, 16, 32 | 8, 16 | 8, 16, 24,32 | 8, 16, 32 | 32 | 32 |
| adresses | 20 | 20, 24 | 20, 24, 32 | 20, 24 | 16, 32 | 16, 24, 32 | 32 | 32 |
| multiplexage | NON | NON | NON | NON | OUI | NON | OUI | OUI |
| interruptions | 8 | 12 | 11 | 8 | NON | 7 | NON | NON |
| multiprocesseur | NON | NON | NON | OUI | OUI | OUI | OUI | OUI |
| synchrone ou asynchrone | S | S | S | A | S | A | S | A |

Tableau 10.20
Principales caractéristiques des bus industriels.

## 2 Liaisons série et parallèle avec périphérique éloigné

Ce type de liaison permet de connecter un châssis à des périphériques plus ou moins éloignés. La liaison sera du type série ou parallèle en fonction du nombre de données à transmettre. La figure 10.21 illustre cette classification dans l'exemple de l'ordinateur personnel.

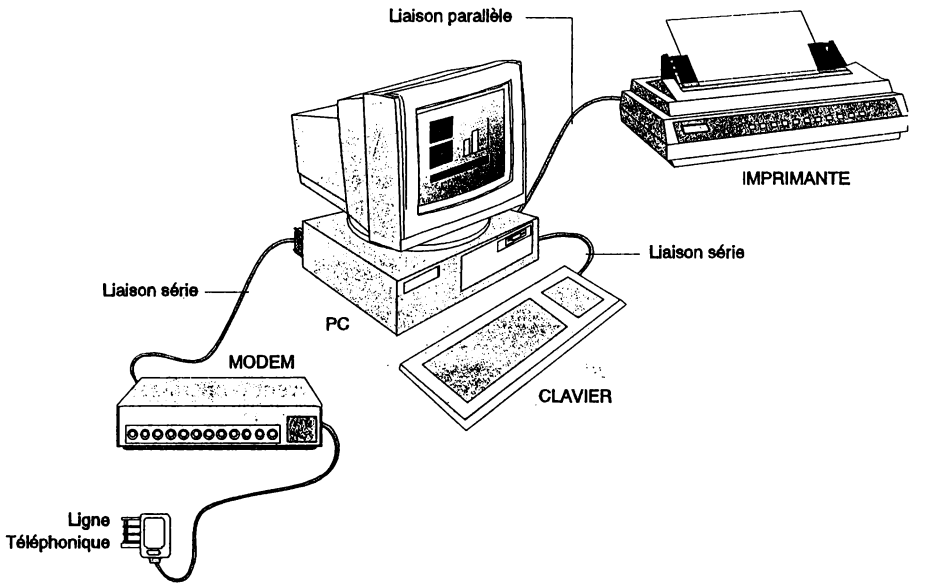



Fig. 10.21
Micro-ordinateur et liaisons.

### *2.1 Liaison série*

Les liaisons série étudiées dans ce paragraphe seront uniquement les très populaires liaisons **RS 232** et **RS 485**.

Généralement utilisée pour transmettre des caractères codés en ASCII (voir le chapitre 1), la liaison série permet de faire dialoguer un processeur et un périphérique lent (clavier, appareil de mesure, etc.). Définie en 1969, elle est un des standards les plus anciens et les plus répandus. C'est une liaison de type série mais un certain nombre de lignes de contrôle portent le nombre de connexions électriques parallèles à 20. Ce chiffre peut paraître énorme pour une liaison de type série. Il faut cependant savoir que toutes les lignes de contrôle ne sont pas toujours utilisées, ce qui entraîne d'ailleurs un certain nombre de difficultés pour les utilisateurs.

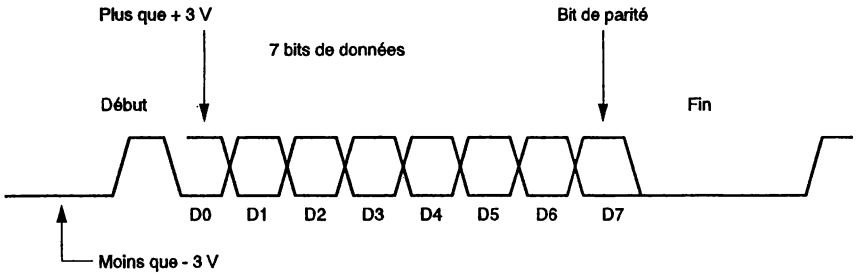



Fig. 10.22
Liaison RS 232.

Conformément au principe de la liaison de type asynchrone, les bits du message sont transmis selon le schéma indiqué figure 10.22. Le message comprend huit bits et deux transitions, l'une pour le début du message et l'autre pour la fin du message. La liaison entre deux équipements est représentée figure 10.23. Toutes les lignes, rappelons-le, ne sont pas toujours utilisées et dans la pratique, de nombreuses versions ont été développées pour la mise en œuvre de ce type de liaison.

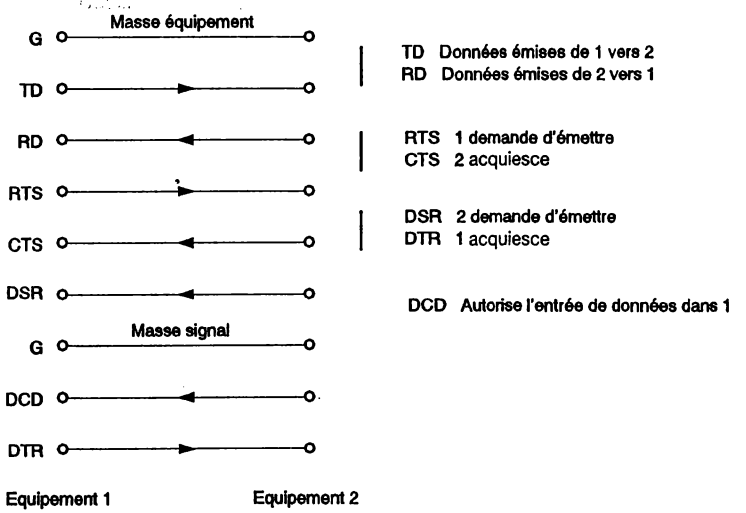



Fig. 10.23
Liaison RS 232 entre deux équipements.

Les communications peuvent être gérées par des circuits spécialisés fonctionnant selon le principe de machines d'états ou par des logiciels implantés dans des microprocesseurs ou des microcontrôleurs. D'autres standards se sont imposés par la suite,

en particulier pour autoriser des débits plus élevés. Le tableau 10.24 ci-dessous résume les caractéristiques principales.

| | RS 232 | RS 423 | RS 422 | RS 485 |
|---|---|---|---|---|
| **Longueur maximum** | 50 m | 1200 m | 1200 m | 1200 m |
| **Débit maximum** | 20 kbits/s | 100 kbits/s | 10 Mbits/s | 10 Mbits/s |
| **Niveaux électriques** | 5 à 15 V | 3.6 à 6 V | ± 2 V min en différentiel | ± 1,5 V min en différentiel |
| **Niveau détection** | 3 V | 0.2 V | ±0,2 V | ± 0,2 V |

Tableau 10.24
Caractéristiques principales des liaisons série.

### *2.2 Liaison parallèle*

Les liaisons précédemment utilisées étaient de type série. Les données étaient donc transmises bit à bit. L'exemple de la liaison RS 232 a cependant montré que les signaux de contrôle étaient transmis simultanément par des lignes supplémentaires. Quand il est nécessaire d'obtenir un débit élevé, le mode parallèle devient indispensable et un certain nombre de bits sont transmis simultanément. Les principaux standards utilisés sont **CENTRONICS**, **SCSI**, **GPIB** ou **IEEE488**. À la différence de la communication par bus fond de panier, la transmission se fait sur des distances assez élevées et des difficultés de synchronisation des bits apparaissent. La figure 10.25 illustre le fonctionnement de la liaison CENTRONICS, utilisée pour relier un ordinateur à une imprimante.

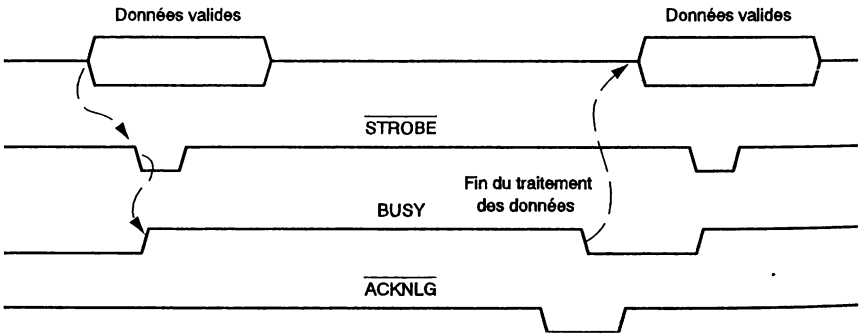



Fig. 10.25
Liaison CENTRONICS.

Le microprocesseur place les données sur les lignes correspondantes et émet un signal d'échantillonnage *STROBE* pour indiquer la validité des données. Le périphérique est alors déclaré actif (*BUSY*). Le périphérique place cette ligne à l'état bas quand les données sont complètement traitées.

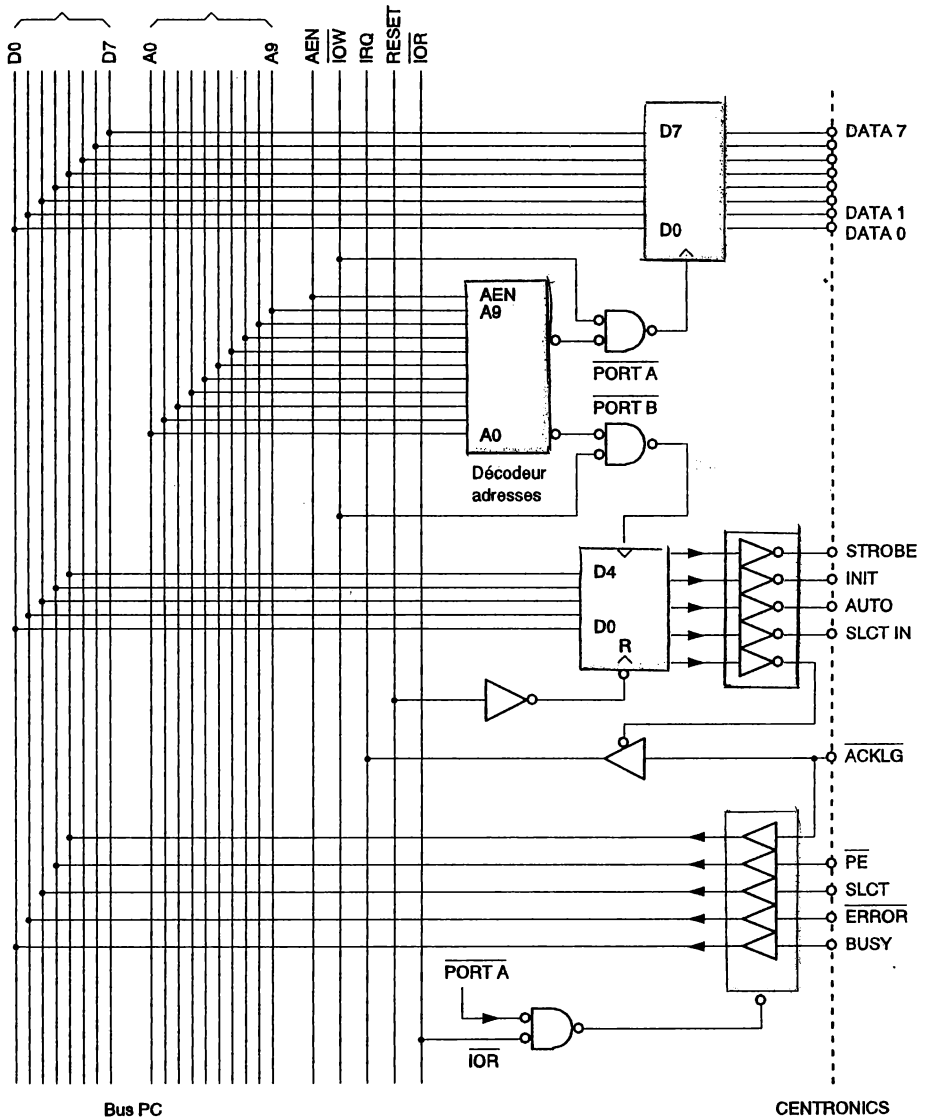



Fig. 10.26
Interface bus PC/CENTRONICS.

Voyant la ligne *BUSY* à l'état bas, le processeur pourra envoyer d'autres données. Un signal d'agrément $\overline{ACKLG}$ peut être utilisé pour déclencher une interruption. Il est actif à l'état bas. D'autres lignes peuvent délivrer des niveaux en provenance du périphérique ou du microprocesseur. Ce sont :

PE : plus de papier

ERROR : impossible d'imprimer,

SCLCT IN : annule la sélection de l'imprimante,

INIT : initialisation de l'imprimante.

La figure 10.26 montre la réalisation matérielle de l'interface entre le bus PC et la liaison CENTRONICS. Pour comprendre les fonctions des différents circuits, il faut remarquer que deux adresses sont affectées au périphérique. La première, *PORT A*, correspond à la lecture de l'état du périphérique (mode lecture) ou à la transmission d'une donnée vers le périphérique (mode écriture). La seconde, *PORT B*, correspond à la configuration du périphérique (mode écriture). Le signal *STROBE* est activé par le programme du microprocesseur.

Les autres protocoles (SCSI et GPIB) seront décrits succinctement. Le standard SCSI (*Small Computer System Interface*) est une liaison parallèle sur 8 bits utilisant les deux modes de transmission : synchrone ou asynchrone. Elle est utilisée pour relier l'unité centrale au disque dur et des circuits spécifiques ont été développés pour gérer la liaison. En 1986, HEWLETT PACKARD décida de définir un protocole pour faire communiquer les instruments de mesure. Ainsi naquit la liaison GPIB. Elle permet de relier des appareils sur une distance de vingt mètres avec un débit de 1Mbyte/s. Le standard correspondant est l'IEEE 488.

## 3 Les réseaux locaux

Il s'agit maintenant de communiquer entre machines ou équipements sur des distances plus élevées, un kilomètre et au-delà. Les liaisons série et parallèle étudiées précédemment ne sont plus adaptées car elles nécessitent des acquittements dans la transmission des groupes de bits qui se traduisent par des allers et retours sur les lignes et des délais prohibitifs quand les distances sont élevées. Il faut alors transmettre des messages sans attendre du récepteur une confirmation de bonne réception. Ce principe général a donné lieu à une grande diversité de standards. Dans tous les cas, le transfert des données se fait en mode série puisque les distances sont importantes. Des livres entiers sont consacrés à la description de ces standards de communication. Il n'est pas question dans cet ouvrage général de se livrer à une telle entreprise. Quelques éléments seront

simplement donnés pour différencier les deux grandes classes : les réseaux déterministes comme FIP, PROFIBUS et CAN, et les réseaux non déterministes comme ETHERNET.

Dans le premier cas, un protocole défini permet à chaque équipement connecté de placer un message sur le câble quand son tour est arrivé. Les autres sont en position d'écoute ou de lecture des données.

Dans le deuxième cas, chaque équipement peut « parler », c'est-à-dire placer des bits sur le câble du réseau quand il le désire. Il faut alors savoir gérer les **collisions** qui apparaissent inévitablement quand plusieurs équipements parlent simultanément

## Les UART (Universal Asynchronous Receivers Transmitters)

Les différents modes de liaison vus précédemment sont, en pratique, mis en service par des circuits spécialisés. Ces circuits permettent de transformer les données présentes sur le bus du microprocesseur en une série de bits se succédant au rythme choisi par l'utilisateur pour réaliser la liaison série adéquate.

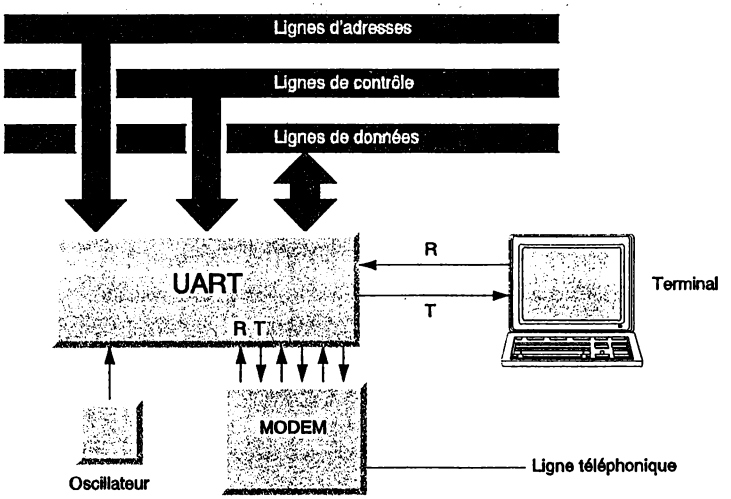



Fig. 10.27
Utilisation d'un UART.

La figure 10.27 illustre une application type : le bus du processeur est en liaison avec un terminal par une liaison série type RS 232 ; il est également en liaison avec une ligne téléphonique via un modem. Ces circuits sont paramétrables. Il est de ce fait possible

de choisir le type de liaison (synchrone ou asynchrone), le débit de la liaison, les contrôles de parité, etc.

Pour choisir la configuration, il suffit d'écrire les mots d'état convenables dans les registres du circuit.

Différents modes de fonctionnement sont possibles comme le montre la figure 10.28 :
- liaison **simplex** : les données sont émises de manière unidirectionnelle du processeur vers le périphérique ou du périphérique vers le processeur ;
- liaison **half duplex** : les données sont émises, soit du processeur vers le périphérique, soit du périphérique vers le processeur, mais pas simultanément ;
- liaison **full duplex** : les données sont émises simultanément dans les deux sens.

La figure 10.29 détaille l'architecture interne d'un USART particulier (le SCN 2651 de PHILIPS).

Ce circuit permet de gérer une liaison en mode synchrone ou asynchrone et offre le choix du débit parmi seize possiblilités. Il permet également de créer les signaux nécessaires à la commande d'un circuit de contrôle d'un modem. Le lecteur reconnaîtra les signaux définis par le protocole RS 232.

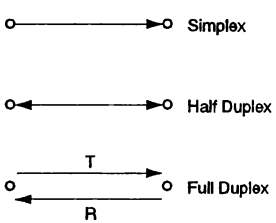
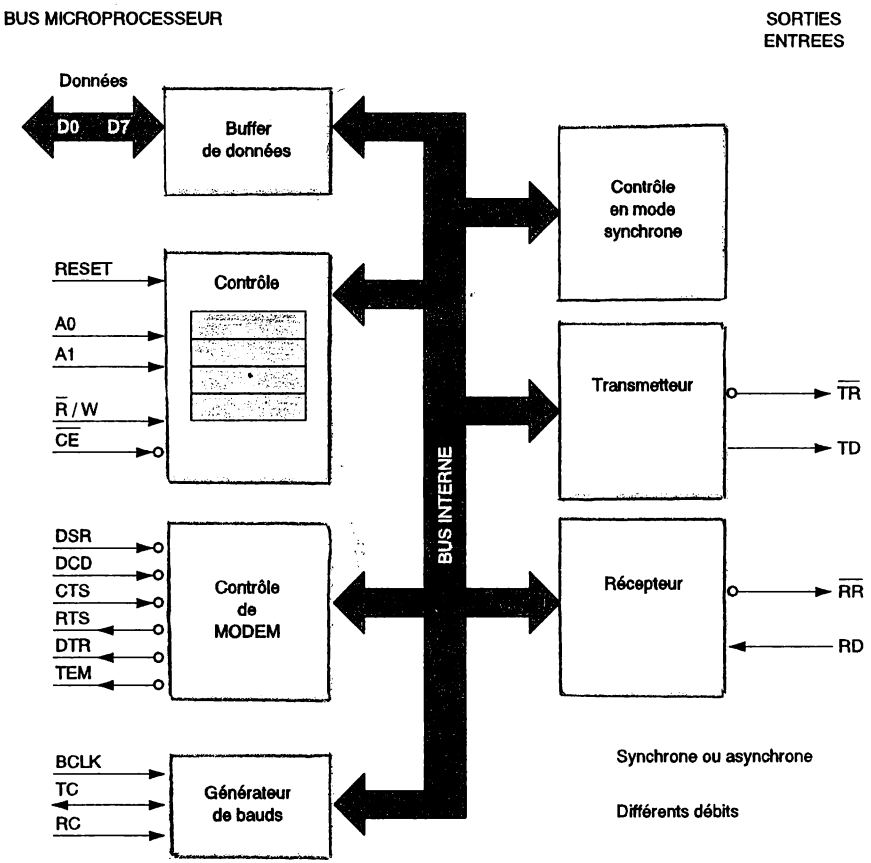



Fig. 10.28
Les modes de liaison.

Ce circuit n'est qu'un exemple parmi une grande variété de circuits disponibles. Certains circuits permettent de gérer jusqu'à huit canaux différents. Il faut noter que ces circuits incorporent des mémoires tampon puisque les débits de données sur le bus du microprocesseur et sur les liaisons ne sont pas forcément compatibles. Il est souvent nécessaire de mémoriser un certain volume d'informations quand le canal de transmission est saturé.




TD Données transmises en série
RD Données reçues en série
$\overline{TR}$ Transmetteur prêt
$\overline{RR}$ Récepteur prêt

$\overline{DSR}$, $\overline{DCD}$, $\overline{CTS}$, $\overline{RTS}$, $\overline{DTR}$ Signaux de contrôle type RS232

$\overline{TEMP}$ Transmetteur vide

$\overline{R}$ / W Ordre de lecture ou d'écriture

BCLK, TC, RC Horloges du générateur de bauds, du transmetteur et du récepteur

Fig. 10.29
Exemple d'un USART.

Chapitre 11

# FILTRES À CAPACITÉS COMMUTÉES ET FILTRES NUMÉRIQUES

## SOMMAIRE


- Limitations dues aux structures classiques
- Intégrateur à capacité commutée
- Filtre à variable d'état
- Limites du système


*L'objectif est de trouver une structure de filtre actif d'ordre 2, passe-haut, passe-bas, passe-bande ou réjecteur se prêtant à l'intégration. Les configurations bâties autour d'un seul amplificateur opérationnel comme le filtre de* ***Sallen et Key*** *sont intéressantes dans le cas où la fonction de transfert est parfaitement choisie : passe-haut par exemple. Le principal intérêt de ce filtre réside dans sa simplicité. Ces structures se prêtent mal à la commutation passe-haut, passe-bas ou passe-bande. Les figures 11.1 et 11.2 représentent deux filtres passe-haut et passe-bas de Sallen et Key.*
*La transformation passe-bas vers passe-haut s'obtient en permutant les résistances et les condensateurs.*

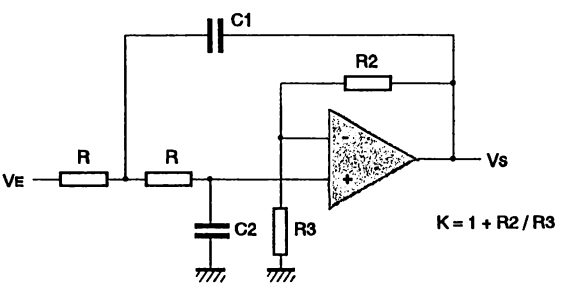



Fig. 11.1
Filtre passe-bas de Sallen et Key.

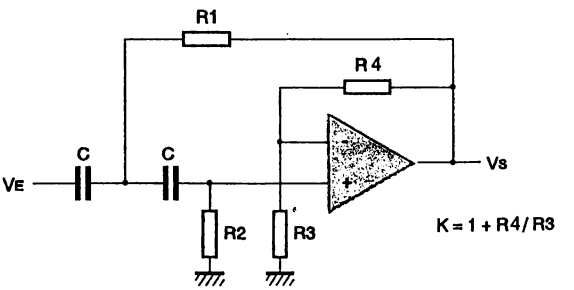



Fig. 11.2
Filtre passe-haut de Sallen et Key.

*Si l'on souhaitait disposer, dans un même circuit intégré, d'un filtre actif passe-haut et d'un filtre actif passe-bas, il faudrait prévoir un système de commutation des éléments passifs. Cette commutation mettrait en jeu un grand nombre de commutateurs analogiques. Pour cette raison, on adopte une configuration d'apparence plus complexe qui regroupe simultanément toutes les fonctions de transfert dont on peut avoir besoin dans un problème de filtrage. D'autre part, cette structure doit pouvoir facilement s'adapter à un type de filtre particulier : Bessel, Butterworth, Tchebycheff, Legendre, etc. C'est dire que l'on doit pouvoir agir simplement et indépendamment sur les deux paramètres principaux :*
*– la fréquence de coupure $\omega_0$,*
*– le coefficient de surtension Q.*

## Limitations dues aux structures classiques

Avec la structure passe-bas de Sallen et Key, figure 11.1, la fonction de transfert vaut :

$$\frac{V_S}{V_E} = \frac{K}{R^2 C_1 C_2 + R[2C_2 + C_1(1-K)]p + 1}$$

On peut donc isoler les deux paramètres *fréquence centrale* et *coefficient de surtension* :

$$\omega_0 = \frac{1}{R\sqrt{C_1 C_2}} \text{ et } Q = \frac{\sqrt{C_1 C_2}}{2C_2 + C_1(1-K)}$$

Avec la structure passe-haut de la figure 11.2, les résultats sont les suivants :

$$\frac{V_S}{V_E} = \frac{KR_1 R_2 C^2}{R_1 R_2 C^2 p^2 + C[2R_1 + R_2(1-K)]p + 1}$$

De la même manière que précédemment, on en déduit fréquence centrale et coefficient de surtension :

$$\omega_0 = \frac{1}{C\sqrt{R_1 R_2}} \text{ et } Q = \frac{\sqrt{R_1 R_2}}{2R_1 + R_2(1-K)}$$

Avec ces deux configurations, il est impossible d'agir sur la fréquence et le coefficient de surtension de manière **indépendante.**

On cherche une structure différente.

Le fonctionnement de l'intégrateur représenté à la figure 11.3 est supposé connu. La fonction de transfert s'écrit :

$$\frac{V_{OUT}}{V_{IN}} = -\frac{1}{RCp}$$

et la constante de temps vaut :

$$\tau = RC$$

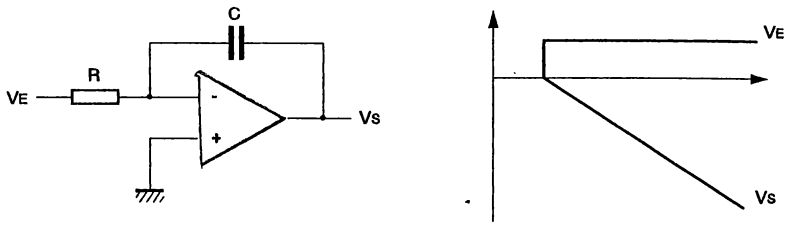



Fig. 11.3
Schéma de l'intégrateur.

On se propose dans un premier temps de montrer qu'il peut être remplacé par le schéma de la figure 11.4.

## Intégrateur à capacité commutée

Le schéma de l'intégrateur à capacité commutée est représenté à la figure 11.4.

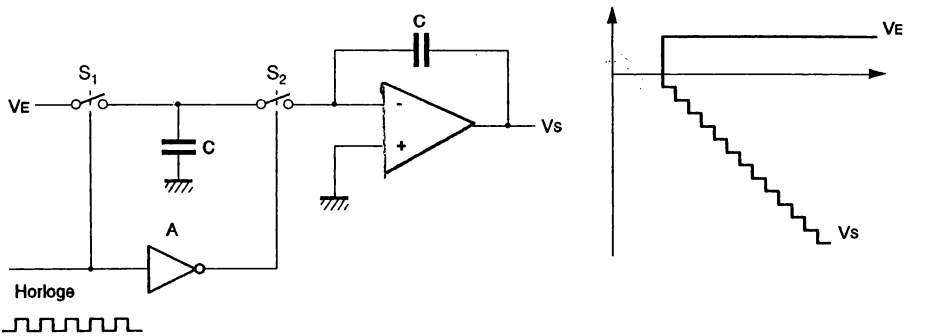



Fig. 11.4
Schéma de l'intégrateur à capacité commutée.

Les interrupteurs $S_1$ et $S_2$ sont alternativement fermés par le signal d'horloge à la fréquence $F_h$. Lorsque $S_1$ est fermé, $S_2$ est ouvert et le condensateur $C_1$ se charge jusqu'à $V_{IN}$. À la fin de chaque demi-période d'horloge, pendant laquelle $S_1$ est fermé, la charge de $C_1$ vaut :

$$Q(C_1) = V_{IN} \cdot C_1$$

Lorsque le signal d'horloge change d'état, dernière demi-période d'horloge, $S_1$ est ouvert et $S_2$ fermé, la charge de $C_1$ est transférée dans $C_2$. La quantité de charge transférée pendant une période d'horloge vaut $V_{in} \cdot C_1$. Dans ce cas le courant traversant $C_2$ vaut :

$$I = \frac{\Delta Q}{\Delta t} = \frac{V_{IN} C_1}{T} = V_{IN} C_1 F_h$$

La résistance équivalente entre l'entrée $V_{IN}$ et l'entrée inverseuse vaut :

$$R = \frac{V_{IN}}{I} = \frac{1}{C_1 F_h}$$

Ce qui signifie que $S_1$ et $S_2$, lorsqu'ils sont actionnés par le signal à la fréquence horloge, sont équivalents à la résistance $R$ de la figure 11.3. La constante de temps $\tau = R \cdot C_2$ vaut finalement :

$$\tau = \frac{C_2}{C_1} F_h$$

L'intérêt de cette structure réside dans sa possibilité d'intégration dans un circuit intégré.

Dans ce cas, on remplace une résistance fixe, peu précise, par une résistance variable, de valeur connue avec précision. La constante de temps de l'intégrateur dépend du rapport des deux valeurs de capacité. Celles-ci étant fabriquées sur la même puce, leurs valeurs sont faciles à contrôler et à bien maîtriser. Cette caractéristique donne finalement à l'intégrateur des caractéristiques précises et stables vis-à-vis de la température.

## Filtre à variable d'état

Deux intégrateurs et un sommateur suffisent pour obtenir le filtre, dit à variable d'état, de la figure 11.5.

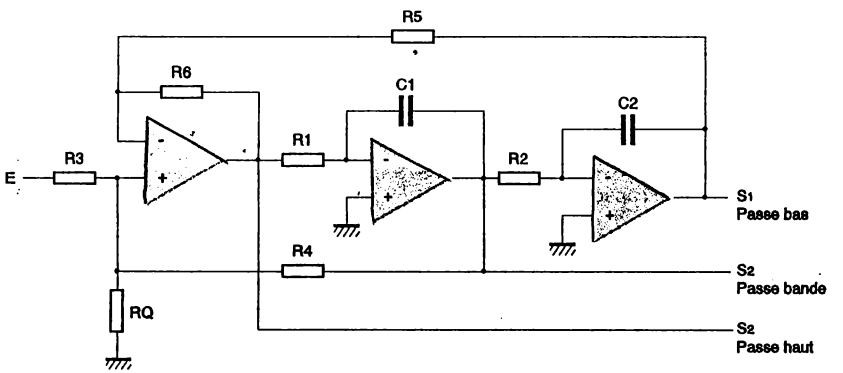



Fig. 11.5
Filtre à variable d'état.

### 1 Calcul de la fonction de transfert du filtre à variable d'état

$$V_{S1} = -A \times \frac{V_{S1}}{R_1 R_2 C_1 C_2 p^2} + B \times V_E - C \times \frac{V_{S1}}{R_1 C_1 p}$$

avec :

$$A = \frac{R_6}{R_5}$$

$$B = \frac{R_4 R_Q}{R_4 R_Q + R_3 R_4 + R_3 R_Q} \times \frac{R_6 + R_5}{R_5}$$

$$C = \frac{R_3 R_Q}{R_3 R_Q + R_3 R_4 + R_4 R_Q} \times \frac{R_6 + R_5}{R_5}$$

Ce qui donne :

$$V_{S1}\left[1 + \frac{A}{R_1 R_2 C_1 C_2 p^2} + \frac{C}{R_1 C_1 p}\right] = BV_E$$

$$\frac{V_{S1}}{V_E} = \frac{B}{A} \times \frac{R_1 R_2 C_1 C_2 p^2}{\frac{1}{A} R_1 R_2 C_1 C_2 p^2 + \frac{C}{A} R_2 C_2 p + 1}$$

et on en déduit :

$$\omega_0 = \sqrt{\frac{R_6}{R_5}} \times \sqrt{\frac{1}{R_1 R_2 C_1 C_2}} \quad \text{et} \quad Q = \frac{A}{C R_2 C_2 \omega_0}$$

Soit :

$$Q = \frac{1 + \frac{R_4}{R_3} + \frac{R_4}{R_Q}}{1 + \frac{R_6}{R_5}} \times \sqrt{\frac{R_6}{R_5}} \times \sqrt{\frac{R_1 C_1}{R_2 C_2}}$$

Cette structure est aussi appelée biquadratique car les fonctions de transfert sont des rapports de deux polynômes du second degré. Le schéma du filtre de la figure 11.5 met en œuvre deux intégrateurs $U_2$ et $U_3$. Pour des raisons de simplification et de clarté, on a représenté deux résistances $R_1$ et $R_2$. Dans la pratique, il s'agit bien d'une capacité commutée, comme dans le schéma de la figure 11.4, équivalente à une résistance $R_1$ ou $R_2$. Pour ce filtre, on calcule les trois fonctions de transfert correspondant au :

- $S_3/E$ : passe-haut,
- $S_2/E$ : passe-bande,
- $S_1/E$ : passe-bas.

$$S_3/E = K \frac{p^2/\omega_0^2}{p^2/\omega_0^2 + 1/Qp/\omega_0 + 1}$$

avec :

$$\omega_0 = \sqrt{\frac{R_6}{R_5}} \times \sqrt{\frac{1}{R_1 R_2 C_1 C_2}}$$

et :

$$Q = \frac{1 + \frac{R_4}{R_3} + \frac{R_4}{R_Q}}{1 + \frac{R_6}{R_5}} \times \sqrt{\frac{R_6}{R_5}} \times \sqrt{\frac{R_1 C_1}{R_2 C_2}}$$

Les fonctions de transfert $S_2/E$ et $S_1/E$ s'obtiennent à partir de $S_3/E$ :

$$S_2/E = -\frac{1}{R_1 C_1 p} S_3/E$$

soit :

$$S_1/E = \frac{R_5}{R_6} K \times \frac{1}{p^2/\omega_0^2 + 1/Qp/\omega_0 + 1}$$

Quel est l'intérêt pratique de cette configuration ?

Si nous posons les hypothèses simplificatrices suivantes :
$R_1 = R_2 R$,
$C_1 = C_2 C$.
$R_3 = R_4$.
$R_5 = R_6$.
les paramètres du filtre deviennent :

$$\omega_0 = \frac{1}{RC} = 2\pi F_0$$

et :

$$Q = 1 + \frac{R_4}{2R_Q}$$

La notion de filtre à variable d'état apparaît clairement : la résistance $R$, et donc la fréquence horloge $F_h$, agissent directement et **uniquement** sur la fréquence de coupure $F_0$. Les résistances $R_4$ et $R_Q$ agissent **uniquement** sur le coefficient de surtension $Q$. L'intérêt de ce filtre réside donc dans la possibilité de calage sur une fréquence stable et précise puisqu'il est possible de générer numériquement une fréquence $F_h$ stable, précise et indépendante de la tension ou température. Ceci n'est évidemment pas le cas si la capacité commutée est remplacée par des résistances $R_1$ et $R_2$.

## 2 Filtre réjecteur

En additionnant les sorties $S_1$ et $S_3$, on obtient un filtre réjecteur dont la fonction de transfert vaut :

$$V_{REJ}/E = K\frac{p^2/\omega_0^2+1}{p^2/\omega_0^2+1/Q\cdot p/\omega_0+1}$$

Ce filtre présente un zéro de transmission à la fréquence $F_0$.

## 3 Limites du système

En présence d'un système échantillonné, le spectre du signal de sortie a l'allure représentée à la figure 11.6.

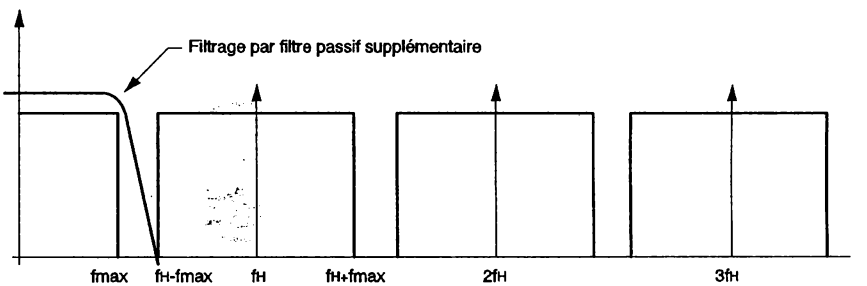



Fig. 11.6
Spectre du signal de sortie d'un filtre à capacités commutées.

Ce spectre se compose du signal utile compris par exemple entre 0 et $F_{max}$ et de bandes de modulation centrées autour de $F_h$, $2F_h$, $3F_h$, etc. et comprises entre $F_h - F_{max}$ et $F_h + F_{max}$. Dans une utilisation pratique, $F_h$ vaut 50 à 100 fois $F_{max}$. Les signaux centrés autour de $F_h$, $2F_h$, $3F_h$, etc., devront être éliminés au moyen d'un filtre passif dont les tolérances n'ont que peu d'importance, et dont la fréquence de coupure se situera entre $F_{max}$ et $F_h - F_{max}$.

Ce filtre passif élimine les composantes haute fréquence et améliore le rapport signal/bruit. Les filtres à capacités commutées peuvent être employés jusqu'à quelques centaines de kHz. Ceci entraîne des fréquences d'échantillonnage de plusieurs MHz. Les limitations en fréquence proviennent essentiellement des commutateurs fonctionnant à la fréquence $F_h$. Pour travailler avec des fréquences $F_{max}$ plus importantes, on est donc tenté de limiter le rapport $F_h/F_{max}$ pour rester dans les limites d'utilisation des commutateurs. Ceci se traduit par un filtrage plus difficile des bandes latérales autour de $F_h$. Pour faciliter le filtrage, on est tenté d'augmenter le rapport $F_h/F$

Des valeurs de 50 à 100 sont de bons compromis habituels dans les circuits intégrés spécialisés accomplissant cette fonction.

### 3.1 *Circuits intégrés regroupant un ou plusieurs filtres à capacités commutées*

| circuit | fréquence maximale | rapport $F_{max}/F_h$ |
|---|---|---|
| **MF10** | 30 kHz | 50 ou 100 |
| **MAX 260** | 7,5 kHz | |
| **MAX 261** | 57 kHz | programmable par micro |
| **MAX262** | 140 kHz | |
| **ML2110** | 30 kHz | 50 ou 100 |
| **ML2111** | 150 kHz | 50 ou 100 |

ENTRAÎNEMENT

**1** Calculez la fonction de transfert d'un filtre défini de la manière suivante :

$$\frac{VP}{VE} = \frac{VPBAS}{VE} - \frac{VPBANDE}{VE} + \frac{VPHAUT}{VE}$$

Examinez successivement la réponse en amplitude et la réponse en phase.
Que peut-on en déduire ?
Établissez le schéma de principe à partir d'un filtre à capacités commutées.

Chapitre 12

# CARTE D'ACQUISITION VIDÉO POUR PC

## SOMMAIRE

- Définition du signal vidéocomposite
- La numérisation vidéo
- Utilisation des adresses

*Cette carte est prévue pour être installée dans un PC avec 1 Mo de mémoire vive et une carte écran VGA.*

*Le schéma synoptique général de la carte de numérisation et d'acquisition d'image est représenté figure 12.1.*

*Le signal vidéo original est un signal vidéocomposite PAL ou SECAM, ou composé des signaux indépendants R, V, B et synchronisation.*

*En permanence, une trame sur deux du signal vidéo est mémorisée. La mémorisation peut être interrompue sur ordre de l'opérateur via le clavier du PC. Après l'interruption de mémorisation, le contenu des mémoires n'est pas altéré et la dernière image mémorisée est toujours présente en RAM.*

*Pour chaque pixel de chaque ligne, on dispose d'une information définissant la couleur et la lumière de ce pixel. Le contenu de la mémoire n'est qu'un fichier de données définissant chaque point. Il est donc intéressant de transférer ce fichier, premièrement dans la mémoire centrale du PC, et deuxièmement sur un support magnétique, disque ou disquette, pour archivage ou traitement.*

*A plusieurs instants, le fichier image se situe dans la mémoire centrale du PC. Il est alors intéressant de profiter des possibilités de calcul de l'appareil.*

*Des modifications peuvent être envisagées et elles aboutiront, par exemple, au compactage du fichier image, ou à la modification de formes ou de couleurs. Des traitements sans modification peuvent aussi être envisagés et des fonctions comme le zoom ou la détection de formes ou de couleurs apparaissent. Finalement le fichier image pourra, après traitement et modification, être transféré dans la mémoire écran originale du PC. Dans ce cas, la modification est inévitable car les données doivent être compatibles avec les standards d'affichage VGA*

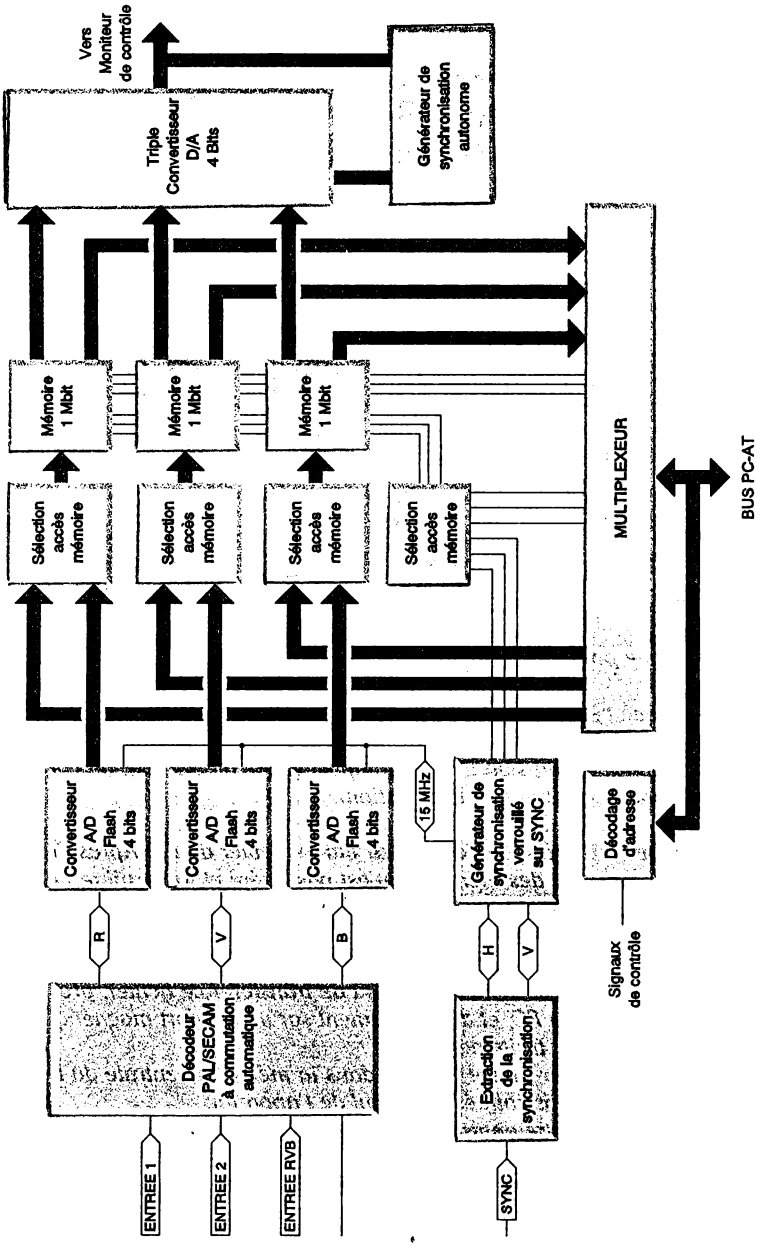



Fig. 12.1
Schéma synoptique de la carte d'acquistion d'image.

## Définition du signal vidéocomposite

Un tube d'analyse d'image délivre les trois signaux primaires : (*R* rouge), (*V* vert) et (*B* bleu).

On définit un signal de luminance *Y* qui représente la lumière du point :

$$Y = 0{,}30 \cdot R + 0{,}59 \cdot V + 0{,}11 \cdot B$$

Si $R = V = B = 0$, $Y = 0$, ce niveau correspond au niveau du noir.

Si $R = V = B = 1$, $Y = 1$, ce niveau correspond au niveau du blanc.

On définit ensuite les signaux de **différence de couleur** :

En PAL :

$$U = 0{,}493(R - Y)$$

$$V = 0{,}877(B - Y)$$

Les signaux *U* et *V* modulent, en amplitude, une sous-porteuse à 4,43361875 MHz. La modulation est du type à porteuse supprimée. Pour plus d'informations, on se reportera au chapitre 8 du tome 1.

En SECAM :

$$D'_R = -1{,}9\,(R - Y)$$

$$D'_B = 1{,}5\,(B - Y)$$

Alternativement, une ligne sur deux, $D'_R$ et $D'_B$, modulent en fréquence deux sous-porteuses à 4,250 MHz et 4,40625 MHz.

Dans les deux cas, le signal de synchronisation composite, ligne et trame, est un signal logique à deux états. Le niveau haut se confond avec le niveau du noir.

Le signal vidéocomposite résulte de l'addition des trois signaux suivants :

- signal de synchronisation composite,
- signal de luminance,
- sous-porteuse modulée en amplitude ou en fréquence, PAL ou SECAM.

## La numérisation vidéo

Trois solutions différentes sont envisageables pour la numérisation d'un signal vidéo. Ces trois solutions diffèrent par le type de signal numérisé :

- signal vidéocomposite,
- signal de luminance et signaux différence de couleur,
- signaux primaires *R*, *V*, *B*.

Un micro-ordinateur traite uniquement les signaux *R*, *V* et *B*. Seule la troisième solution est envisageable. Les sources vidéo composites extérieures sont dans les formats suivants :
– signal vidéocomposite,
– signaux *R*, *V* et *B* dans le cas des caméras.

Pour que cette carte puisse être utilisée dans la plupart des cas, elle doit s'adapter aux formats les plus fréquents et être compatible avec le format d'affichage des micro-ordinateurs.

En conséquence, la numérisation doit s'effectuer sur les signaux primaires.

Pour numériser les signaux *R*, *V*, *B*, on a recours à un système dont le schéma synoptique est représenté à la figure 12.2. On suppose premièrement que le signal est vidéocomposite codé PAL ou SECAM ou deuxièmement que l'on dispose des signaux *R*, *V*, *B*, synchronisation. Le sous-ensemble noté décodeur PAL/SECAM est donc chargé de la reconstitution des signaux primaires *R*, *V*, *B*. Éventuellement, il accepte directement des signaux *R*, *V*, *B* extérieurs.

Les signaux résultants du décodage PAL ou SECAM sont immédiatement codés en binaire par trois convertisseurs.

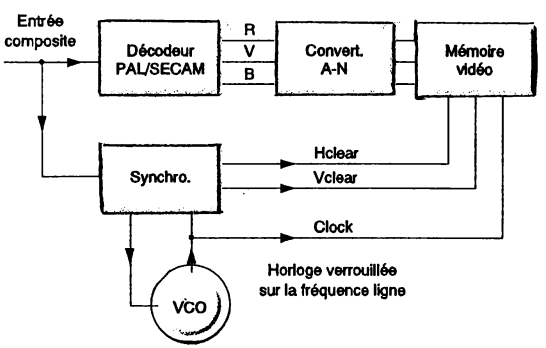



Fig. 12.2
Numérisation des signaux primaires *R*, *V*, *B* pour stockage en vidéo RAM.

Après conversion, les échantillons peuvent être rangés en mémoire.

L'organisation interne des mémoires vidéo 1 Mbit est 306 lignes de 960 pixels, chaque pixel étant codé sur 4 bits.

Sur le schéma synoptique de la figure 12.2, l'écriture ou lecture en mémoire est cadencée par trois signaux notés Hclear, Vclear et Clock. Les convertisseurs A/N sont cadencés par le signal Clock. Le signal Hclear est le signal de synchronisation ligne, il est directement lié au signal vidéocomposite d'entrée, sa période est de 64 µS ($f$ = 15 625 Hz).

Le signal Vclear est lié au nombre de 25 images par seconde ou 50 demi-images par seconde. Vclear a donc une période de 20 ms.

Seule la fréquence du signal Clock reste à définir en sachant que la ligne mémoire a une capacité maximale de 960 pixels, et que dans le signal vidéo, la ligne utile a une durée voisine de 52 µS. La figure 12.3 résume les cas les plus intéressants.

| Fréquence d'échantillonnage | Nombre de bits par couleur | Définition sur une ligne utile de 52 µs | Nombre de boîtiers 1 Mbits CXK 1206 M |
|---|---|---|---|
| 15 MHz | 4 | 780 | 3 |
| | 5 | | 4 |
| | 8 | | 6 |
| 3 fsc 13,300 MHz | 4 | 691 | 3 |
| | 5 | | 4 |
| | 8 | | 6 |
| 4 fsc 17,734 MHz | 4 | 922 | 3 |
| | 5 | | 4 |
| | 8 | | 6 |
| 18,75 MHz | 4 | 975* | 3 |
| | 5 | | 4 |
| | 8 | | 6 |

* 960 sur une ligne utile de 51,2 µs

Fig. 12.3
Récapitulatif de quelques configurations d'échantillonnage *R, V, B.*

Au schéma synoptique de la figure 12.2, on note que le signal clock est verrouillé sur le signal de synchronisation ligne. Le tableau de la figure 12.3 regroupe quatre cas intéressants pour le signal d'horloge. Les quatre fréquences particulières traitées sont : 15 MHz, 3*Fsc*, 4*Fsc* et 18.75 MHz avec :

$$F_{sc} = (1\,135/4 + 1//625) \cdot F_h$$

et $$F_h = 15{,}625 \text{ kHz (fréquence ligne)}$$

Dans chacun des cas, on peut affecter 4, 5 ou 8 bits par couleur. Le nombre de boîtiers mémoire 1 Mbit résultant de cette opération vaut alors 3, 4 ou 6. Dans le cas où l'on affecte 4 ou 5 bits à chaque couleur, chaque pixel est alors codé sur 12 ou 15 bits (2 octets). Dans le cas où chaque couleur est codée sur 8 bits, trois octets sont nécessaires à la définition du pixel. Rappelons que l'on souhaite stocker une demi-image, une trame sur deux dans un système entrelacé.

## 1 Décodeur PAL/SECAM

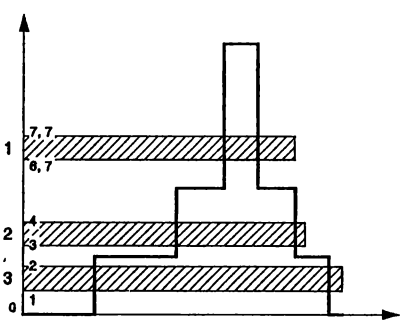
L'intégration a déjà permis la réalisation d'un décodeur PAL/SECAM grâce à un seul circuit intégré. Dans ce cas, certains composants comme la **ligne à retard** chrominance 64 µS et la ligne à retard luminance 300 nS à 500 nS restent externes. En outre, certains circuits doivent être accordés – principalement en SECAM.

Le circuit TDA 8390 est le circuit principal de décodage, le TDA 8490 est le complément pour transformer un décodeur PAL en décodeur PAL/SECAM. Les deux circuits en technologie P2CCD assurent les fonctions de filtrage et ligne à retard. Le circuit TDA 8451 est dédié à la fonction ligne à retard chrominance 64 µS, quant au circuit TDA 8452 il regroupe la ligne à retard luminance, le filtre réjecteur chrominance sur la voie luminance et le filtre passe-bande chrominance.

## 2 Schéma de principe de la carte de numérisation et acquisition

Le schéma de la carte PC a été scindé en sept schémas élémentaires.

### *2.1 Détection de synchronisation*




(1) Seuil du comparateur BURST

(2) Seuil du comparateur LIGNE (BLANKING)

(3) Seuil du comparateur TRAME

Fig. 12.4
Signal Sandcastle et seuils de détection.

Le schéma de principe de la circuiterie de synchronisation d'entrée est représenté à la figure 12.5 (pages 336 et 337). Le signal vidéocomposite est injecté à la broche 11 du TDA 2595, $U_1$ est chargé du tri de la synchronisation. Ce circuit délivre un certain nombre de signaux exploités par les téléviseurs. Dans la configu-

ration actuelle, seuls trois signaux sont exploités : le signal de **synchronisation ligne** issu de la broche 4, le signal de **synchronisation trame** issu de la broche 9 et le signal de **clamp** (ou burst) issu de la broche 6. L'impulsion, dite de clamp, est une impulsion qui permet de réaligner le signal vidéo composite.

Si le signal vidéo composite traverse un condensateur de liaison, le niveau du noir est perdu. Le niveau moyen est fonction du contenu de l'image. Il est donc nécessaire de fixer pendant un bref intervalle de temps, au début de la ligne, le niveau de référence : le niveau du noir.

Ces trois signaux participent à deux actions principales : fabrication d'un signal dit **sandcastle**, à trois niveaux, et verrouillage du générateur universel de synchronisation $U_4$, SAA 1101.

### 2.2 *Élaboration du signal sandcastle*

Le signal issu de la broche 6 correspond au signal de clamp. La présence de la résistance $R_{10}$ placée entre l'alimentation et l'entrée 2 décale le niveau de l'impulsion délivrée à la broche 6. Les deux portes, TTL $U_{33F}$ et $U_{33E}$, mettent l'impulsion en forme pour donner une impulsion positive de + 12 V.

Du générateur de synchronisation universel, on extrait la synchronisation ligne – broche 22 de $U_4$ – et la synchronisation trame – broche 21 de $U_4$ –. Ces deux signaux ont une excursion de + 0,5 V. Ces trois signaux sont pondérés ($R_{82}$, $R_{83}$ et $R_{88}$) et envoyés vers un additionneur à diodes $D_2$, $D_3$ et $D_4$. Aux bornes de $R_{83}$, on récupère un signal sandcastle à trois niveaux dont l'aspect est représenté à la figure 12.4 (voir page précédente).

### 2.3 *Verrouillage du signal de synchronisation*

Les signaux de synchronisation ligne et trame issus des broches 4 et 9 du TDA 2595 – $U_1$ – pilotent le générateur universel SAA 1101 qui se verrouille sur les signaux incidents. Ce circuit est un PLL, la fréquence de sortie est un multiple de la fréquence d'entrée (fréquence ligne).

Avant son injection vers le générateur universel de synchronisation, les deux signaux de synchronisation subissent une mise en forme : $C_6$ et $R_{15}$ pour la synchronisation ligne, $C_4$ et $R_8$ pour la synchronisation trame. Après mise en forme, le signal de synchronisation ligne devient un signal d'effacement ligne utilisé pour la reconstitution du signal sandcastle et pour l'asservissement en ligne du SAA 1101 recevant ce signal à la broche 12.

Le front montant de l'impulsion trame est dérivé de manière à obtenir une brève impulsion de remise à zéro des compteurs ligne. Cette impulsion est envoyée sur la broche 12 du SAA 1101.

L'oscillateur du SAA 1101 est une porte inverseuse connectée entre les broches 5 et 6. Cette porte associée aux divers éléments tels que $T_1$, $C_{10}$, $C_{11}$, $C_{106}$ et $D_1$ constitue un VCO. Lorsque le générateur de synchronisation est verrouillé sur les signaux ligne et trame incidents, la fréquence de l'oscillateur vaut exactement $960 \cdot F_h$ soit 15,000 MHz.

Les signaux de sortie notés CLOCKO, HO et VO, du SAA 1101 cadenceront le système d'écriture en mémoire. Ce système d'écriture recevra en outre un signal d'autorisation en synchronisme avec l'image. De cette manière, le début de l'image, première ligne en haut de l'écran, se trouvera au début de la mémoire, premières adresses.

### *2.4 Élaboration du signal GEL*

Le signal d'autorisation d'écriture, en fait un WE pour la mémoire, est élaboré à partir de trois signaux. Seule une des deux trames peut être stockée ; la sélection de la trame est confiée à la bascule $U_{28A}$. Le signal de sortie à la broche 1 bascule à chaque trame, il a donc une période de 40 ms. Ce signal est combiné au signal de synchronisation trame, de manière à interdire l'écriture pendant le retour trame. Ce mécanisme est impératif car une trame comporte 312.5 lignes et la mémoire ne stocke que 306 lignes. Ceci n'altère en rien la qualité de l'image, la trame utile ne comptant que 285 lignes apparaissant effectivement à l'écran.

À la figure 12.5 (pages 336 et 337), les signaux de commande notés CDEGEL pour commande de gel et RSTGEL pour reset gel sont issus du PC. L'utilisateur par pression sur certaines touches autorisera le gel de la prochaine demi-image ou interdira celui-ci. D'un point de vue électrique, CDEGEL et RSTGEL sont normalement à l'état bas. À l'initialisation, une impulsion doit être envoyée sur RSTGEL. Le signal GEL passe alors à l'état bas pendant toute la période utile – 285 lignes – de la même trame. Chaque nouvelle demi-image, une fois sur deux, est écrite en mémoire. Si le niveau de l'entrée CDEGEL passe à 1, l'utilisateur demande le gel, la prochaine trame sera stockée et l'écriture désormais interdite. Pour ce faire, le signal CDEGEL aura une durée minimale de 40 ms et dès la fin de la trame mémorisée, le signal GEL passe à l'état haut, interdisant une nouvelle écriture. Le signal GEL correspond au signal WE des mémoires vidéo. Le retour au mode normal s'effectue en envoyant une brève impulsion à l'entrée RSTGEL.

### *2.5 Schéma du décodeur PAL/SECAM*

Il est représenté à la figure 12.6 (pages 338 et 339). Il regroupe les quatre circuits intégrés Philips participant à la fonction de décodage TDA 8390, TDA 8490, TDA 8451 et TDA 8452. Les deux entrées vidéocomposite sont envoyées vers les broches 14 et 16

du circuit intégré TDA 8452. Lorsque la tension appliquée à la broche 9 est inférieure à 1,5 V, la première entrée – broche 14 – est sélectionnée et lorsque cette tension est supérieure à 4,0 V, la deuxième entrée est sélectionnée. Le schéma de la figure montre les diverses possibilités offertes : sélection d'un signal, signal vidéocomposite parmi 2 ou sélection des entrées *R*, *V*, *B*, synchronisation.

Le circuit TDA 8452 combine filtrage et retard. Le signal de luminance est disponible à la broche 3, le signal de chrominance à la broche 5 et un signal composite retardé à la broche 4. Dans le cas du PAL, luminance et chrominance sont directement exploitées par le TDA 8390 qui, accompagné de la ligne à retard 64 µS, TDA 8451, décode les signaux et restitue les composantes primaires *R*, *V* et *B*.

Les trois potentiomètres $R_{42}$, $R_{43}$ et $R_{44}$ affectent respectivement le contraste, la lumière et la saturation des couleurs.

En SECAM, le signal retardé est légèrement amplifié puis filtré par un filtre en cloche : $R_{57}$, $C_{58}$, $C_{59}$ et $T_2$. Le filtre est ajusté sur la fréquence de 4,286 MHz.

La démodulation des signaux de différence de couleur est assurée par le TDA 8490. Le circuit résonnant $T_3$, $C_{35}$, $R_{58}$ est utilisé à la fois pour la détection du SECAM et pour la démodulation. Le signal retardé et amplifié est noté vidéo composite, puis il est envoyé vers le sous-ensemble de tri de synchronisation. Après le tri des tops de synchronisation et verrouillage d'un générateur universel, on élabore le signal dit de sandcastle, signal à trois niveaux nécessaire aux trois circuits TDA 8390, TDA 8490 et TDA 8451.

Les signaux rouge, vert et bleu sont ensuite transmis au sous-système de numérisation.

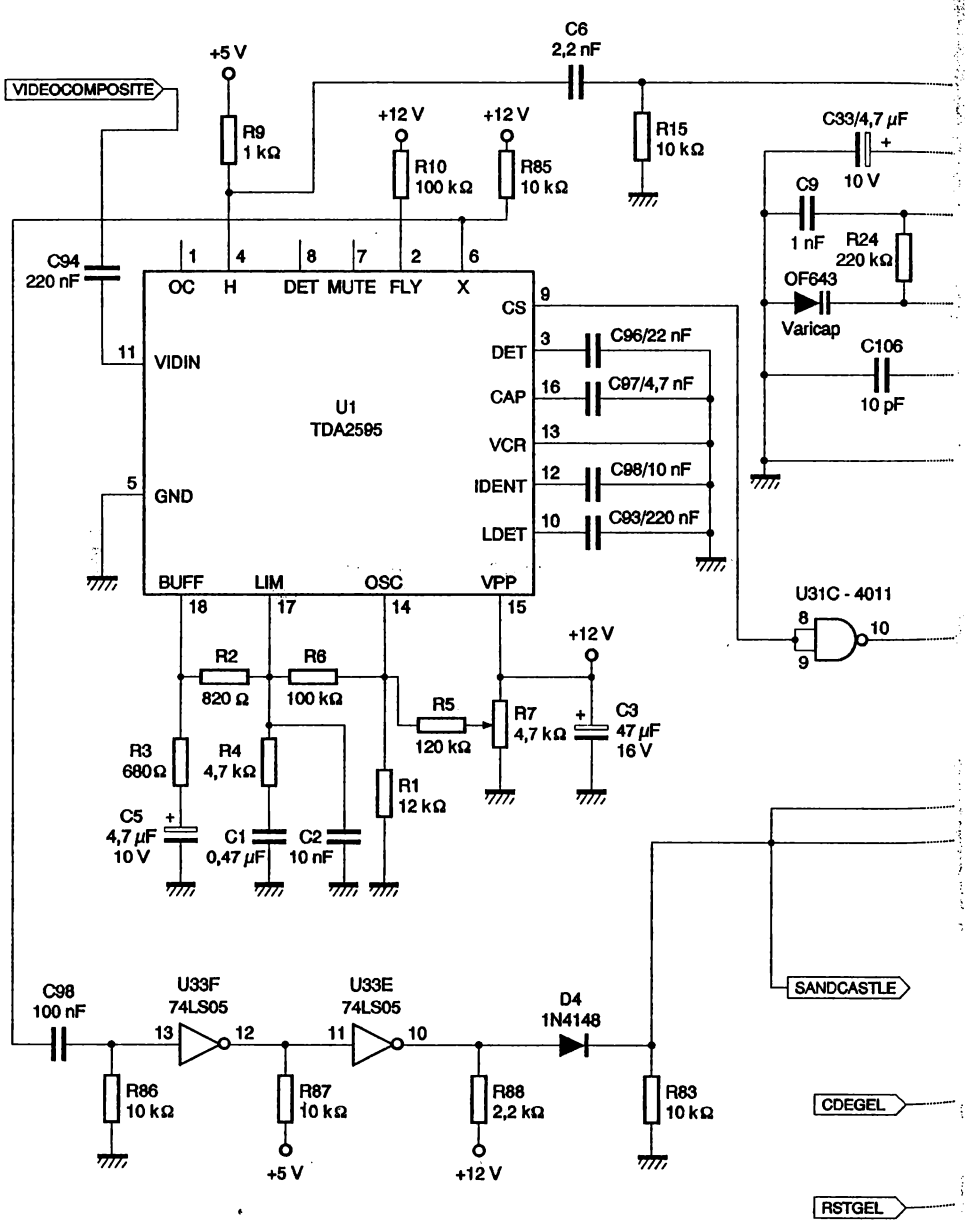



Fig. 12.5
Circuit de synchronisation.

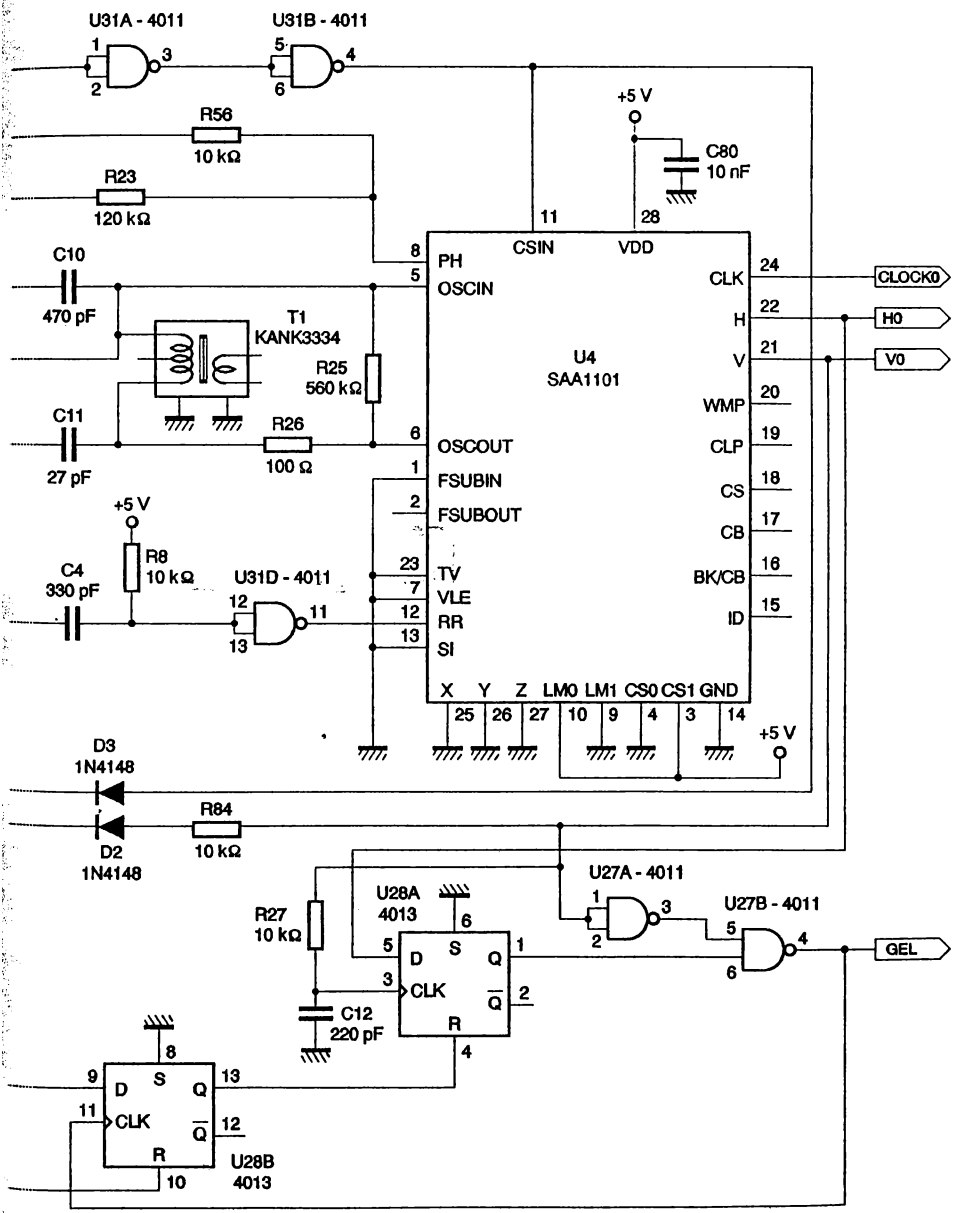



Fig. 12.5 (suite)

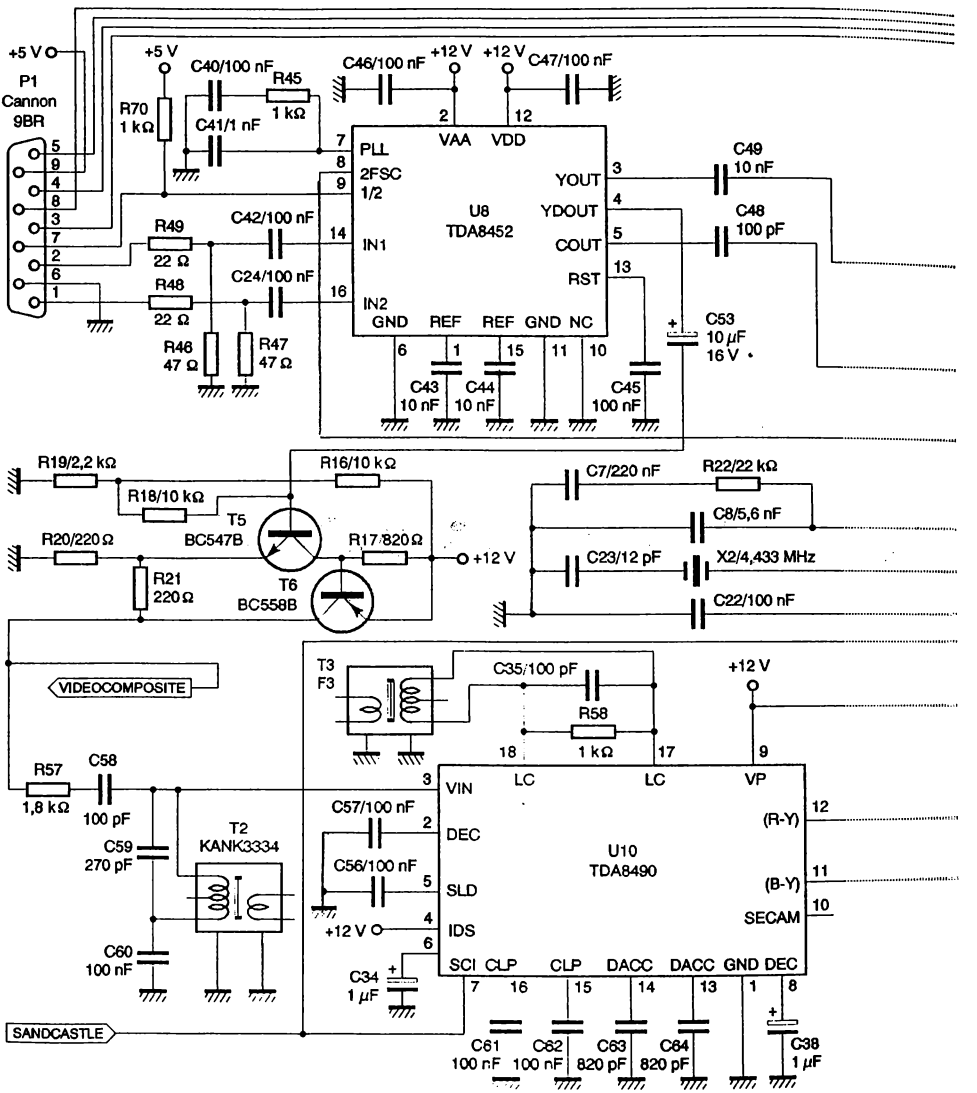



Fig. 12.6
Décodeur PAL/SECAM.

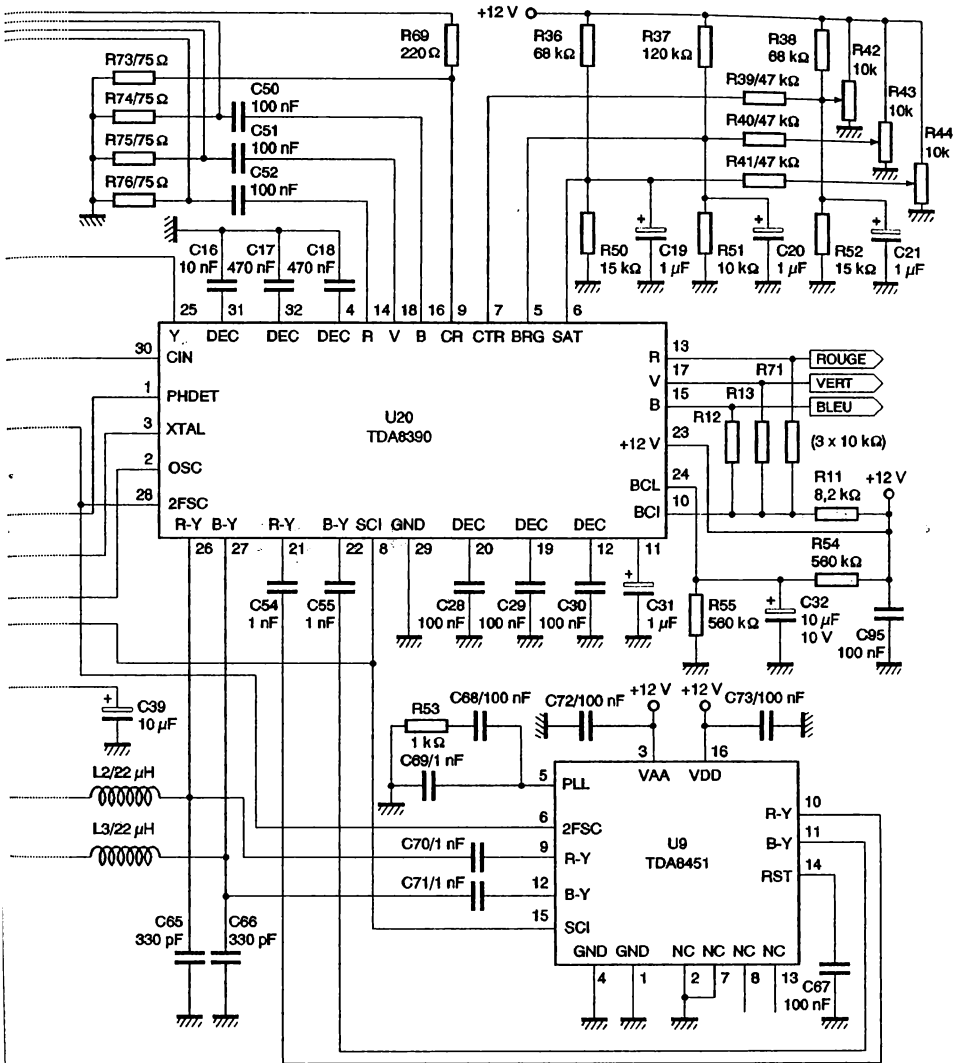



Fig. 12.6 (suite)

## 2.6 *Conversion analogique numérique des signaux R, V, B*

Le schéma de la conversion numérique des signaux analogiques *R*, *V*, *B* délivrés par le décodeur PAL/SECAM est représenté à la figure 12.7. Pour la conversion, on fait appel à trois convertisseurs quatre bits RCA CA3304. Les trois voies *R*, *V* et *B* étant identiques,

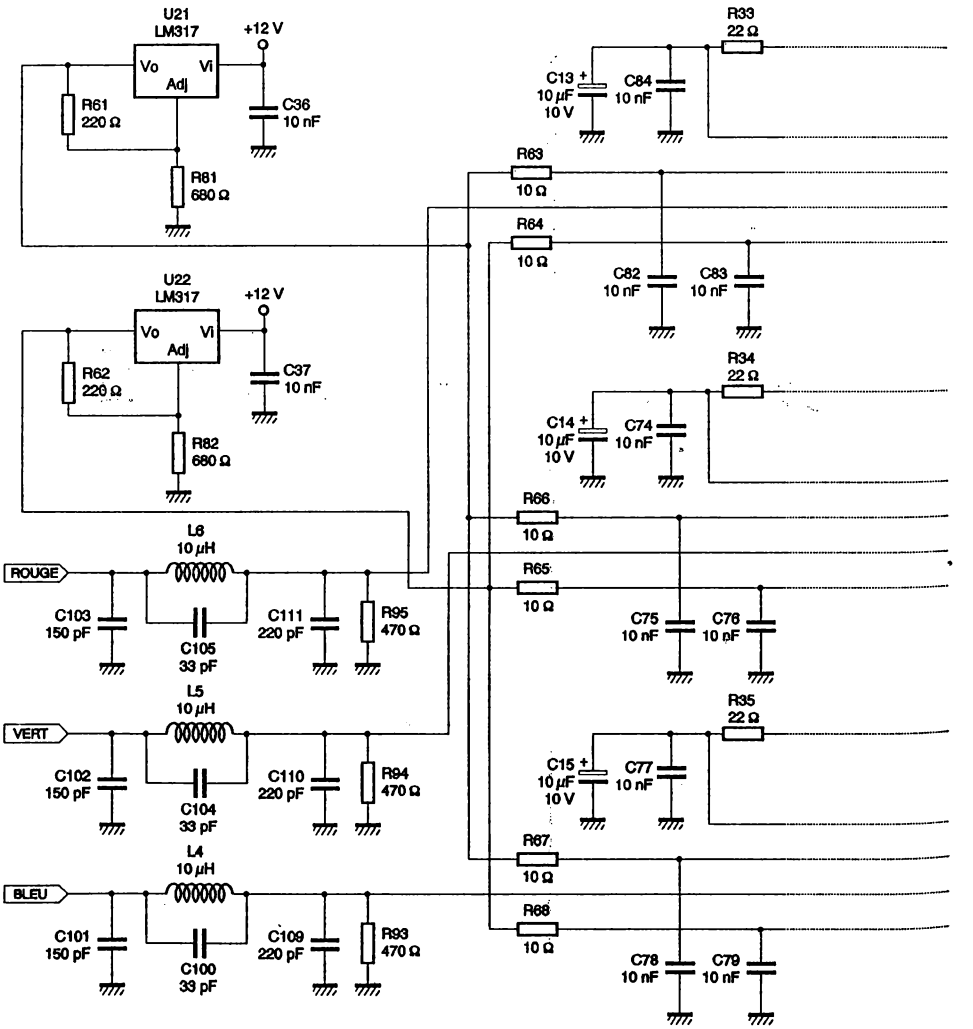



Fig. 12.7
Numérisation *R*, *V*, *B*.

une seule voie sera traitée. Le signal est préalablement filtré avant d'atteindre l'entrée du convertisseur A-D. Ce filtre élimine les phénomènes de **repliement de spectre.**

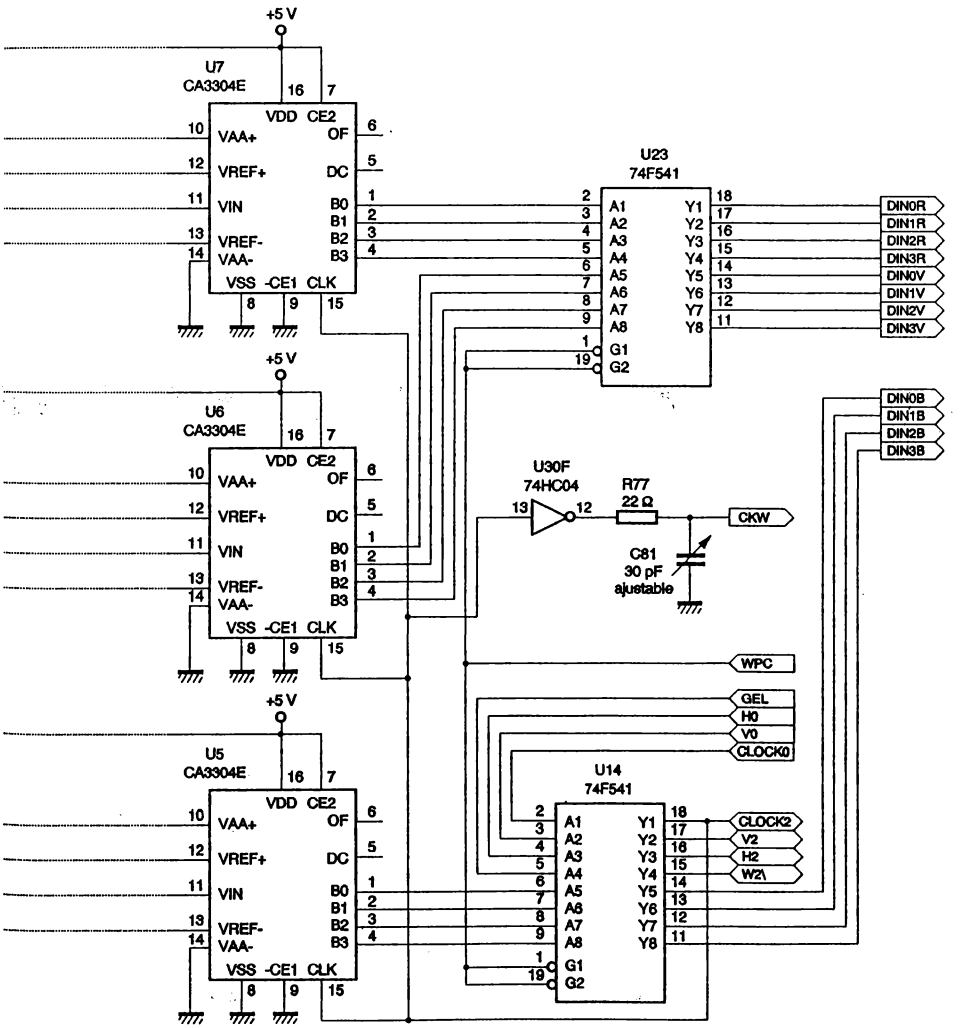



Fig. 12.7 (suite)

L'impédance de sortie du TDA 8390, qui délivre les signaux primaires vaut 50 Ohms, et cette impédance est prise en compte pour calculer le filtre passe-bas d'entrée dont la réponse est donnée à la figure 12.8.

Les tensions des seuils de discrimination sont issues de deux régulateurs $U_{21}$ et $U_{22}$ . Les résistances $R_{81}$ et $R_{82}$ fixent ces tensions. La tension de référence basse est voisine de 2 V, et la tension de référence haute proche de 5 V. Il n'est pas nécessaire que ces tensions soient ajustables, les commandes de lumière et de contraste étant accessibles sur le TDA 8390. La commande de lumière est utilisée pour faire coïncider le niveau du noir avec la tension de référence basse et la commande de contraste est utilisée pour faire coïncider le niveau du blanc avec la tension de référence haute.

Les trois convertisseurs $U_5$, $U_6$ et $U_7$ sont cadencés par le signal d'horloge CLOCK0 en provenance du générateur universel de synchronisation SAA 1101 U4.

Les signaux *R*, *V*, *B* numérisés sur quatre bits sont disponibles aux broches 1, 2, 3 et 4 des convertisseurs. En fonction du niveau du signal WPC, ces signaux sont ou ne sont pas envoyés vers les mémoires. WPC est un ordre venant du micro, si WPC est à zéro, le système d'acquisition – décodeur PAL/SECAM – pilote les mémoires, et si WPC est au + 5 V, le micro pilote les mémoires.

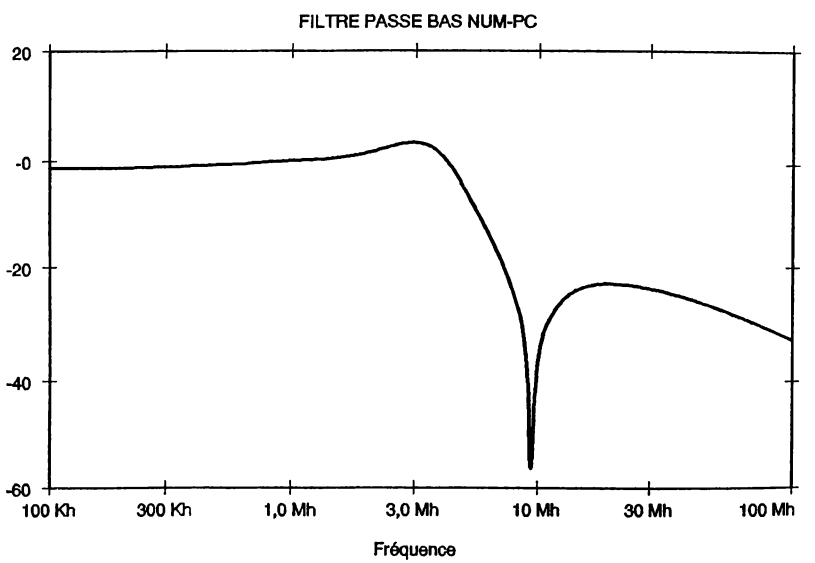



Fig. 12.8
Réponse du filtre passe-bas anti-repliement, entrée des convertisseurs A-D.

## 2.7 Schéma interface PC

Le schéma de l'interface de la carte avec le bus PC/AT est représenté à la figure 12.9 (pages 344 et 345).

Les deux connecteurs situés sur la carte mère ont respectivement 62 et 36 contacts. Sur le schéma de principe, les deux connecteurs notés JP2 et JP3 représentent les deux connecteurs situés en bas de la carte. Ces deux connecteurs s'enfichent dans les supports situés sur la carte mère. Cette carte utilise le bus 16 bits du PC/AT. Les données $D_0$ à $D_7$ se situent sur le connecteur JP2 ainsi que les adresses $A_{10}$ à $A_0$ et les données $D_8$ à $D_{15}$ sur le connecteur JP3 de $C_{11}$ à $C_{18}$. Cette carte s'intègre facilement dans un système existant sans qu'il y ait de conflit avec d'autres cartes d'extension précédemment installées.

La première opération réside donc dans un décodage d'adresse. Pour le 286, l'espace I/O comprend 64 K ports 8 bits ou 32 K ports 16 bits. Dans un PC/AT, certaines cartes utilisent une ou plusieurs adresses d'entrées-sorties. Pour certaines cartes, ces adresses sont bien connues, par exemple les cartes d'extension ports parallèles ou ports série : LPT1, LPT2, LPT3, COM1, COM2 et COM3. Il en est de même pour les cartes vidéo monochrome ou couleur et la carte contrôleur disques.

Avant de choisir une adresse, il est donc impératif de connaître toutes les adresses I/O utilisées par le système. Pour être capable de s'adapter à toutes les configurations, les adresses commençant à $A_0$ jusqu'à $A_{10}$ sont décodées. Les onze bits d'adresses se scindent en deux parties : $A_0$, $A_1$, $A_3$ utilisés pour la carte, et $A_4$ jusqu'à $A_{10}$ dite adresse de base.

L'interrupteur $S_1$ sélectionne l'adresse de base. Cette adresse, codée sur huit bits, peut prendre 255 valeurs différentes de 0 à 255.

Ces valeurs sont comprises entre 000H et 7F8H par pas de 8. On choisira donc une adresse non utilisée et l'exemple de la figure 12.10 (page 346) donne le cas où la carte est adressée en 300H, adresse de la carte prototype. Dans ces conditions, la carte de numérisation et d'acquisition utilise les trois adresses suivantes :
- adresse de base + 0,
- adresse de base + 2,
- adresse de base + 4.

soit dans le cas de l'adresse de base de la carte prototype : 300H, 302H et 304H.

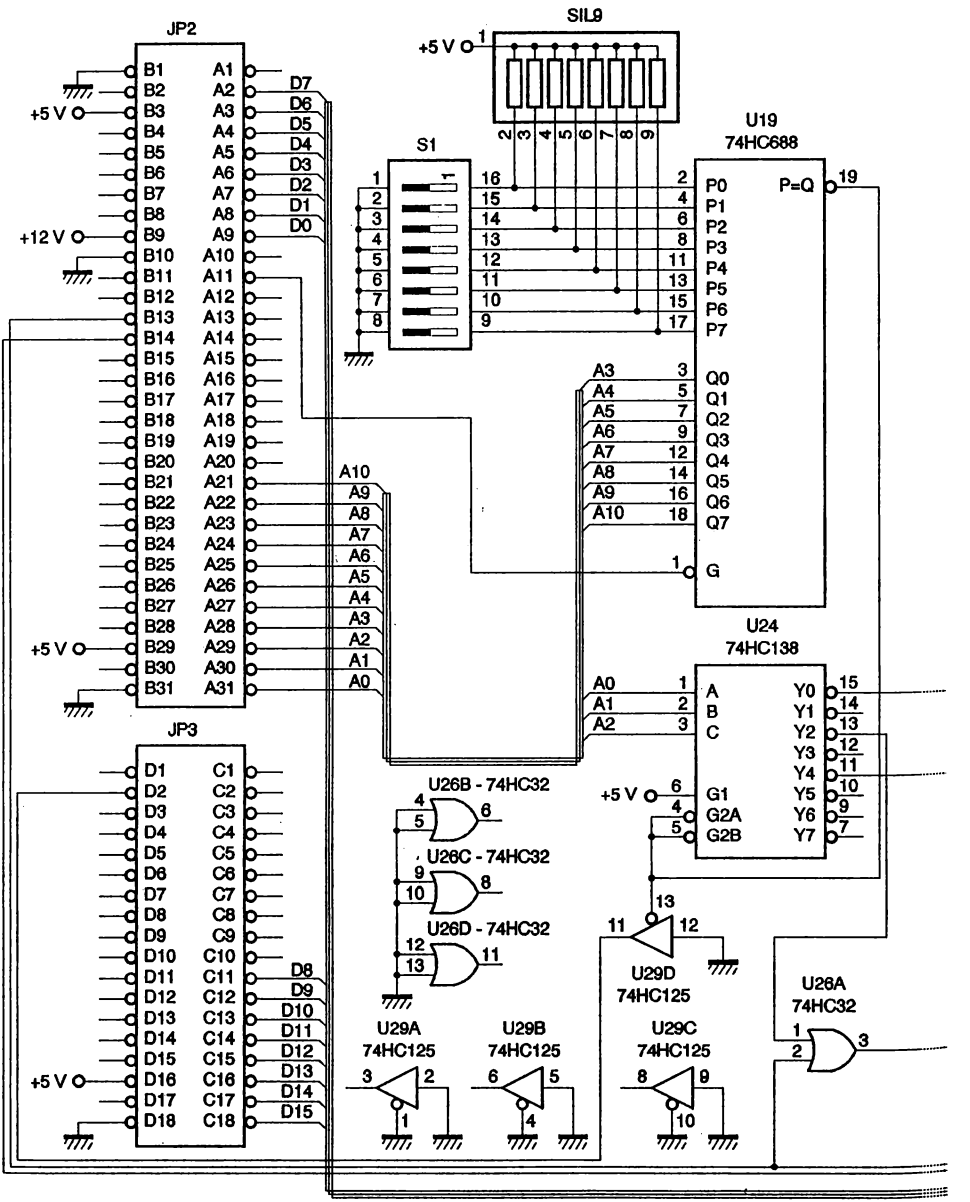



Fig. 12.9
Interface PC.

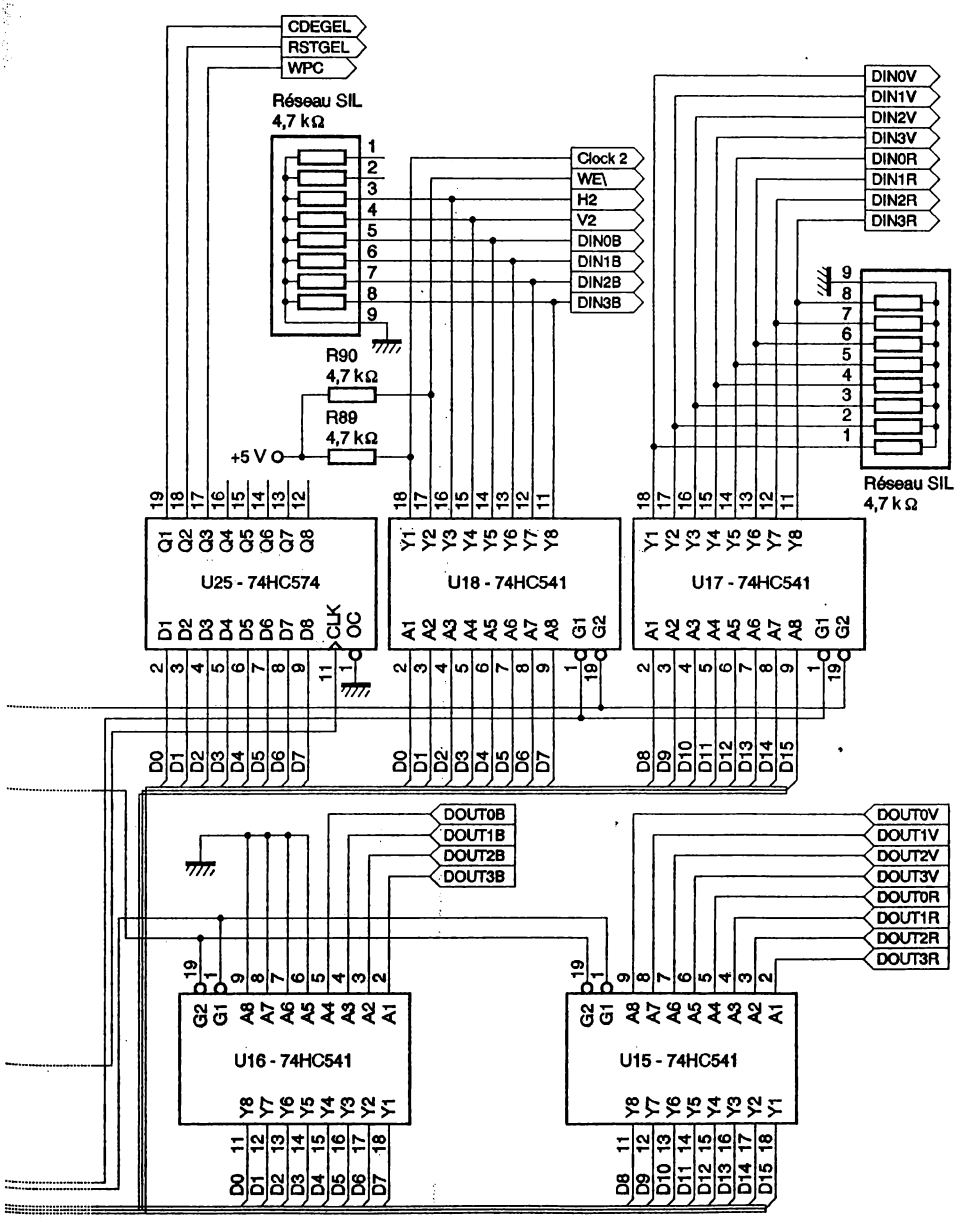
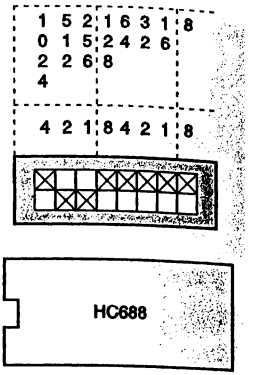



Fig. 12.9 (suite)

| 4 2 1 | 8 4 2 1 | 8 4 2 1 |
|---|---|---|

| A10 | A9 | A8 | A7 | A6 | A5 | A4 | A3 | A2 | A1 | A0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | | | | | | 0 | 1 | 0 |
| | | | | | | | | 1 | 0 | 0 |




Adresse de base la plus basse 000H.
Adresse + 1 008H.
Adresse de base la plus haute 7F8H

Adresse de la carte prototype 300H, 302H, 304H } 3 Adresses utilisées par la carte

Fig. 12.10
Adressage de la carte d'acquisition et numérisation.

## 3 Utilisation des adresses

### 3.1 *Adresse de base + 2*

Cette adresse est utilisée pour stocker, sur la carte, un mot de commande. Pour chacun des signaux, le diagramme des temps est représenté au schéma de la figure 12.11. Seuls les trois bits de plus faible poids sont utilisés.

Le bit de plus faible poids, $D_0$ – broche 19 de $U_{25}$ – est la commande de gel. Cette sortie doit passer au niveau haut pendant au moins 40 ms. Après ce passage à l'état haut, l'écriture est interdite et la mémoire contient la dernière demi-image.

Le bit $D_1$ – broche 18 de $U_{25}$ – est noté RSTGEL pour reset gel. Une brève impulsion permet le retour à la normale après le gel. Chaque nouvelle demi-image, une demi-image sur deux, est écrite dans la mémoire.

Le bit $D_2$ – broche 17 de $U_{25}$ – noté WPC, est utilisé pour le transfert d'un fichier image de l'unité centrale vers la mémoire vidéo de la carte d'acquisition. En régime d'acquisition, WPC est à zéro et en régime de transfert, du PC vers les mémoires SONY, WPC est à l'état haut.

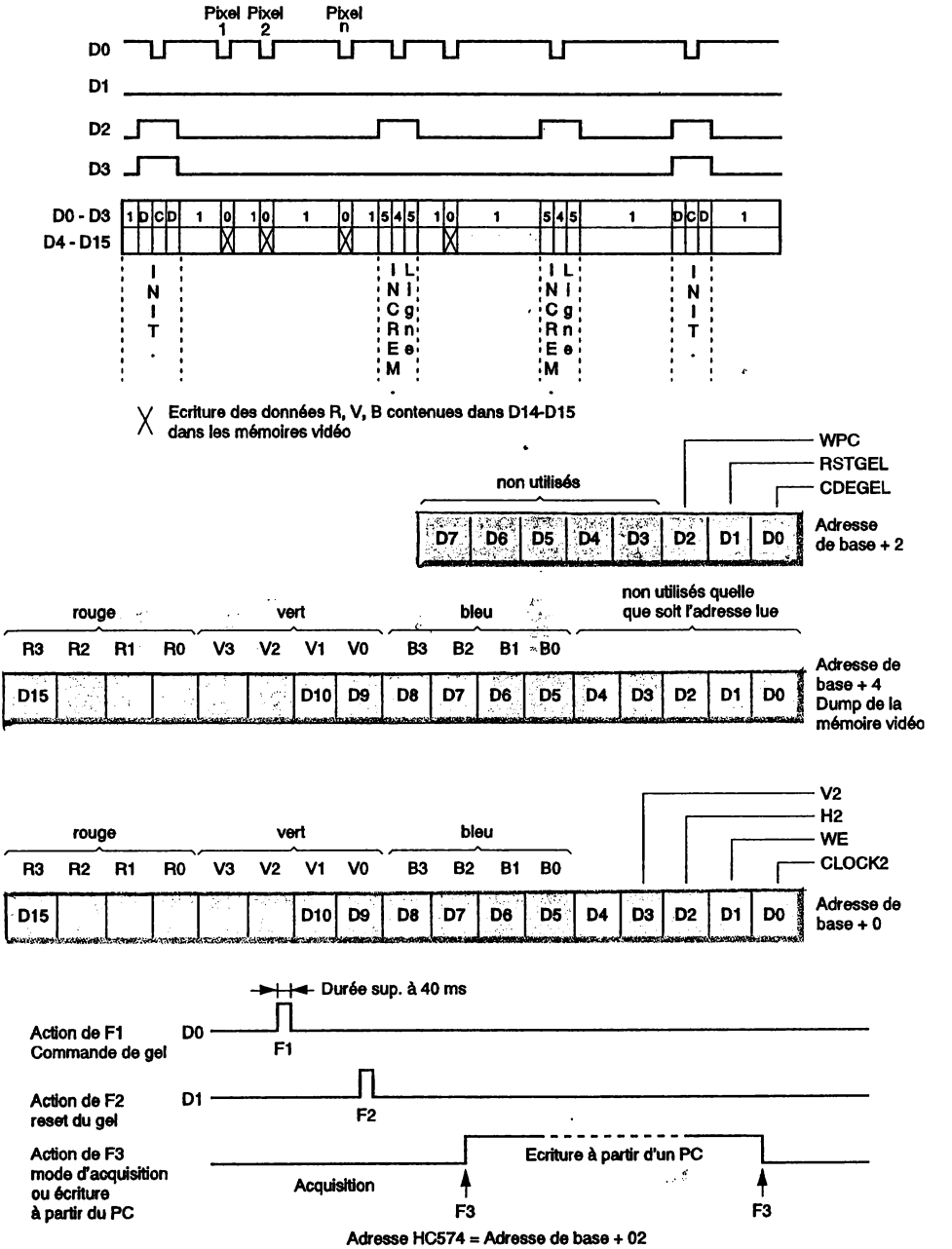



Fig. 12.11

Diagramme des temps des signaux.

### *3.2 Adresse de base + 4*

À l'adresse de base + 4, on trouve une entrée pour le bus du micro. C'est par cette voie que transiteront les informations définissant un **pixel** : 12 bits de données. Le bus de données 16 bits du PC/AT est noté $D_0$ à $D_{15}$, chacune des trois couleurs est codée sur quatre bits.

Les quatre bits les moins significatifs $D_0$ à $D_3$ sont, par câblage, en permanence au niveau 0.

Les quatre bits suivants $D_4$ à $D_7$ reflètent l'intensité du bleu, $D_8$ à $D_{11}$ celle du vert et finalement $D_{12}$ à $D_{15}$ celle du rouge.

D'un point de vue programmation, l'acquisition s'effectue de la manière suivante : lecture de l'information, un coup d'horloge pixel, lecture du pixel suivant, etc. jusqu'à la fin de la ligne, puis ligne suivante jusqu'à épuisement des lignes.

### *3.3 Adresse de base + 0*

Vue du micro, cette adresse est une sortie, et il écrira à cette adresse. À cette adresse, deux opérations différentes sont effectuées. La différence dépend de l'état de $D_1$ correspondant à WE pour les mémoires.

Si $D_1 = 1$, il n'est pas possible d'écrire dans les mémoires et seuls les bits $D_0$, $D_2$ et $D_3$ sont pris en compte. $D_0$ est alors considéré comme horloge pixel en lecture alors que $H_2$ et $V_2$ correspondent à Hclear et Vclear en lecture. Les bits $D_4$ à $D_{15}$ n'ont aucune importance.

Si $D_1 = 0$, le microprocesseur est autorisé à écrire directement dans les mémoires vidéo de la carte d'acquisition et de numérisation. Les bits $D_0$, $D_2$ et $D_4$ concernent alors les compteurs en écriture et les données présentes sur $D_4$ à $D_{15}$ sont inscrites dans les mémoires. Les données sont organisées de la même manière que pour la lecture à l'adresse de base + 4.

## 4 Mémorisation de l'image

La mémorisation des trois composantes *R*, *V* et *B* codées chacune sur quatre bits est confiée à trois mémoires SONY CX 1206M.

Le schéma de principe des sous-ensembles mémoire est représenté à la figure 12.12 (pages 350 et 351).

Ces trois mémoires constituent le cœur de la carte. Ce cœur possède des ramifications avec presque tous les autres sous-ensembles, excepté le décodeur PAL/SECAM directement. Les signaux d'entrée des convertisseurs A-N CA 3304 sont notés DIN de 0 à 3 et le suffixe indique la couleur.

Le cadencement en écriture est assuré par CKW, WE, H2, V2 et CLOCK2.

Les signaux de sortie des mémoires vers le port du PC à l'adresse de base + 4 sont notés DOUT de 0 à 3 et le suffixe indique la couleur. Le cadencement en lecture est assuré par les signaux H2, V2 et CLOCK2. Ces signaux sont les mêmes en écriture et en lecture car on a estimé qu'il était inutile de prévoir les deux opérations simultanées. Finalement, le système de lecture autonome, détaillé dans un prochain paragraphe, fonctionne en mode non entrelacé 624 lignes.

Dans ce schéma, on retiendra que la mémoire peut être vue comme un bloc unique ayant une entrée et deux sorties. Chaque entrée ou sortie regroupe douze bits de données et quatre signaux de gestion de la mémoire.

## 5 Conversion D-A

À tout instant, dans les mémoires vidéo, on peut lire les échantillons. Chacune des 306 lignes numérisées comporte 960 pixels, seuls 780 pixels participent à l'élaboration de la demi-image de 285 lignes véritablement utiles.

Les mémoires SONY ont trois ports indépendants, un port d'écriture et deux ports de lecture. Cette indépendance est temporelle entre les signaux actionnant un système de lecture ou d'écriture. Dans la pratique, ceci signifie que le système de lecture est complètement autonome et qu'il est inutile qu'il soit asservi au système d'écriture.

Le schéma de principe du générateur de synchronisation associé au triple convertisseur D-A est représenté à la figure 12.13 (pages 352 et 353). Ce schéma est assez simple, les deux pièces maîtresses étant un générateur de synchronisation autonome SAA 1101 et le triple convertisseur D-A Brooktree BT103. Un quartz à 15,000 MHz fixe la fréquence de l'oscillateur sur SAA 1101. Le générateur de synchronisation est en mode non entrelacé, la broche 23 étant reliée à la tension d'alimentation positive.

Les signaux notés CLOCK1, H1 et V1 cadencent les compteurs de lecture des mémoires vidéo. La porte $U_{30A}$ inverse le signal d'horloge et le signal inversé est envoyé vers l'entrée horloge du triple convertisseur D-A. Cette inversion est nécessaire car la conversion doit concerner les données utiles et stables en sortie des mémoires. Le diagramme des temps correspondants est donné au schéma de la figure 12.14 (page 354).

Le mot de douze bits représentatif de la couleur et luminosité du pixel transite des mémoires vers le triple convertisseur D-A. La conversion est effectuée par $U_3$ Brooktree BT103Kc et les trois courants de sortie IOR, IOV et IOB développent aux bornes des résistances $R_{30}$, $R_{32}$ et $R_{31}$ des tensions directement utilisables par un moniteur *R, V, B*. Chaque entrée doit avoir une impédance d'entrée de 75 Ohms.

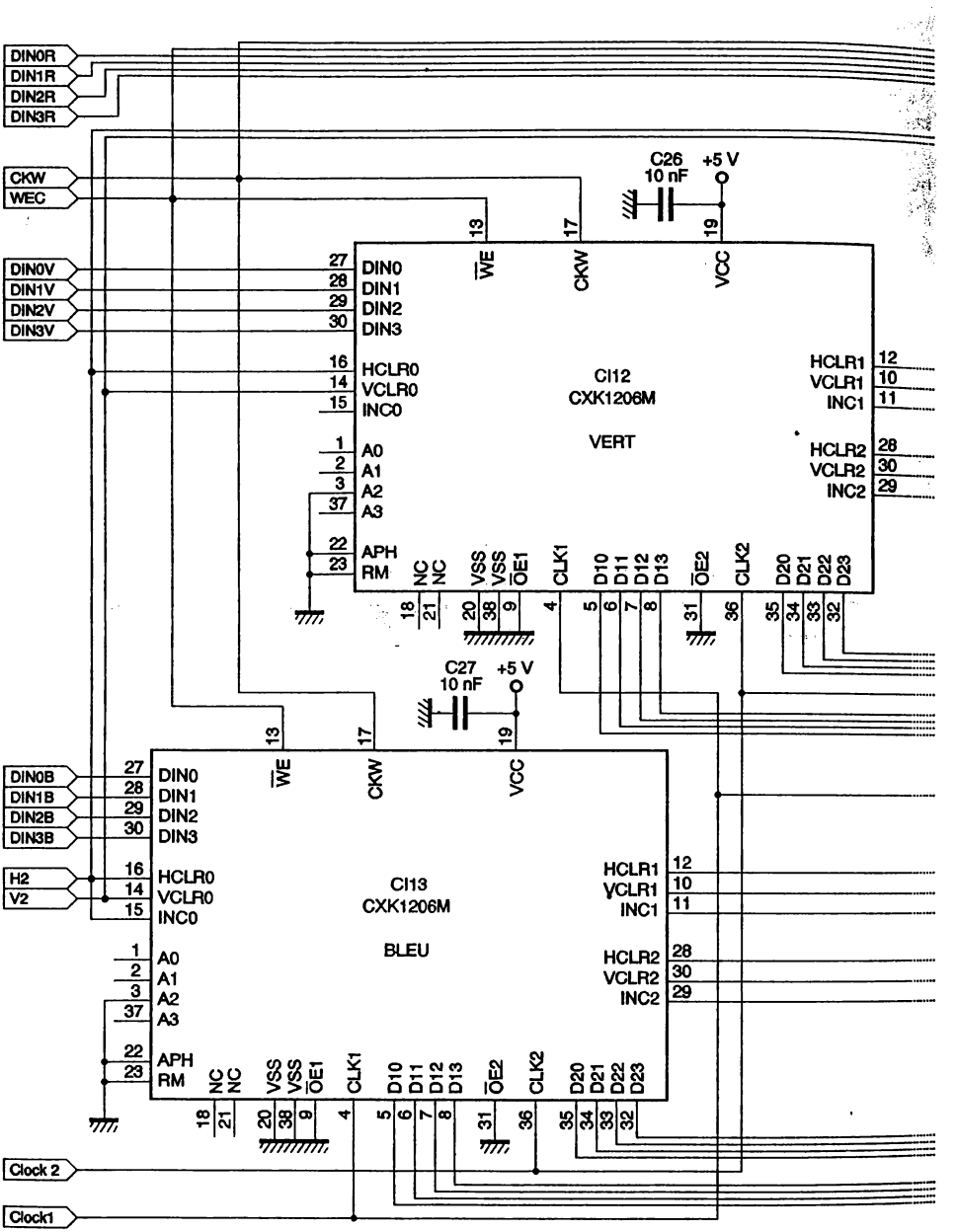



Fig. 12.12

Mémorisation des composantes *R*, *V*, *B*.

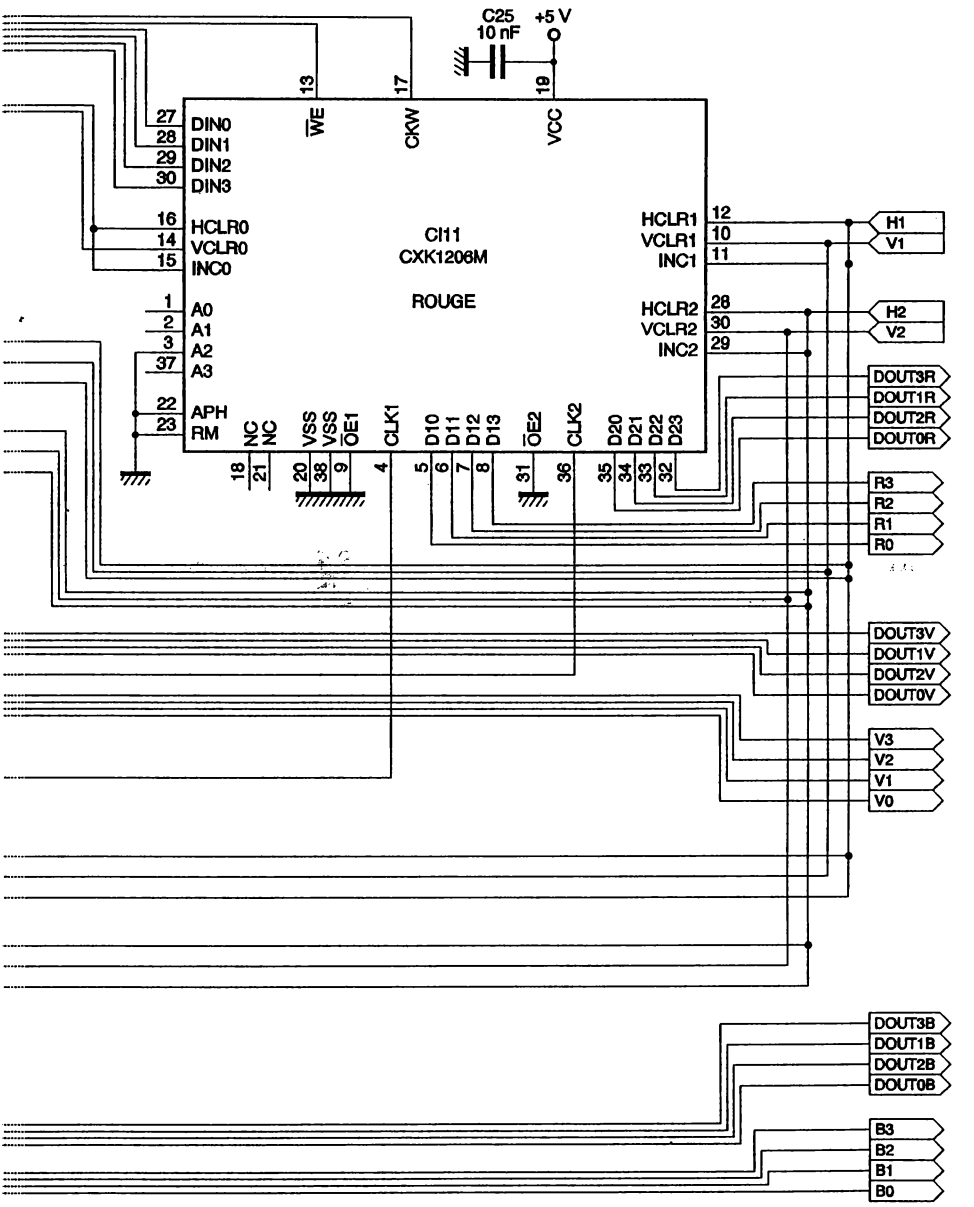



Fig. 12.12 (suite)

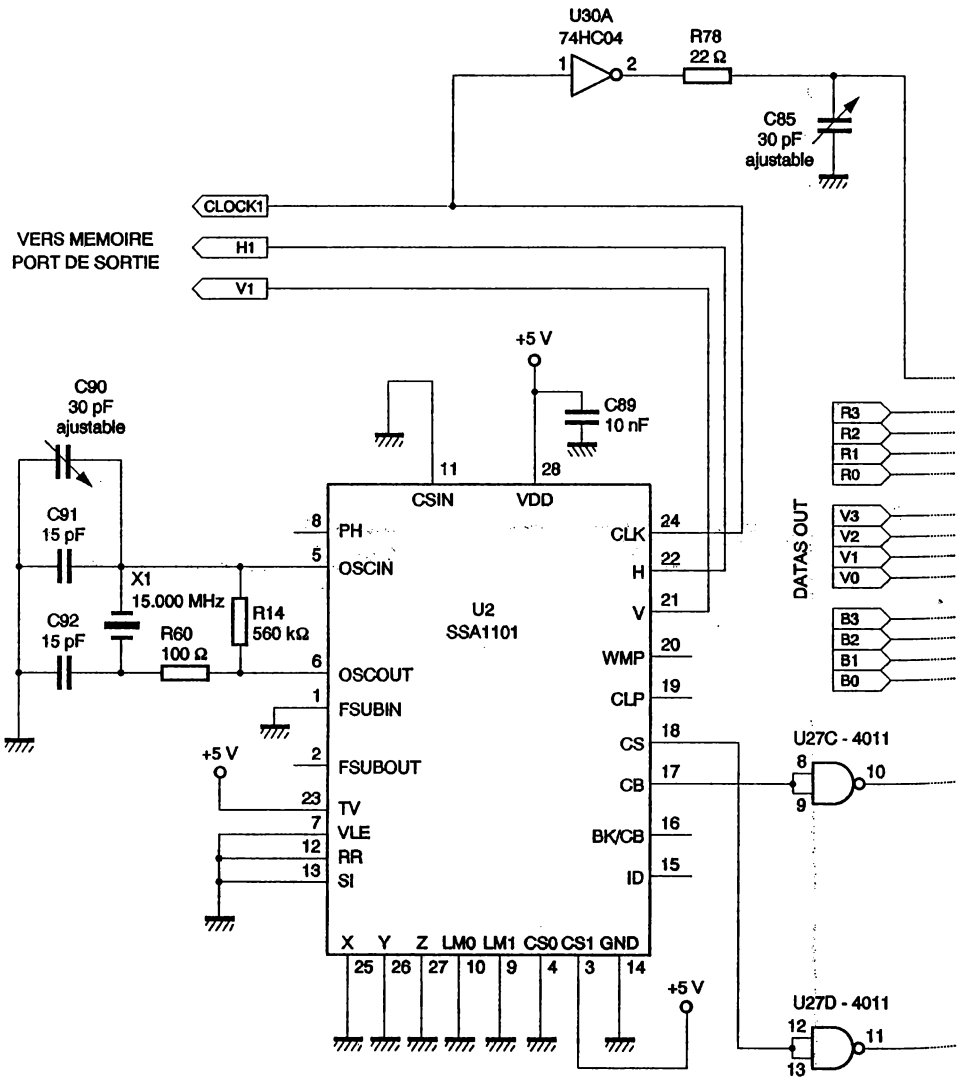



Fig. 12.13
Conversion D-A.

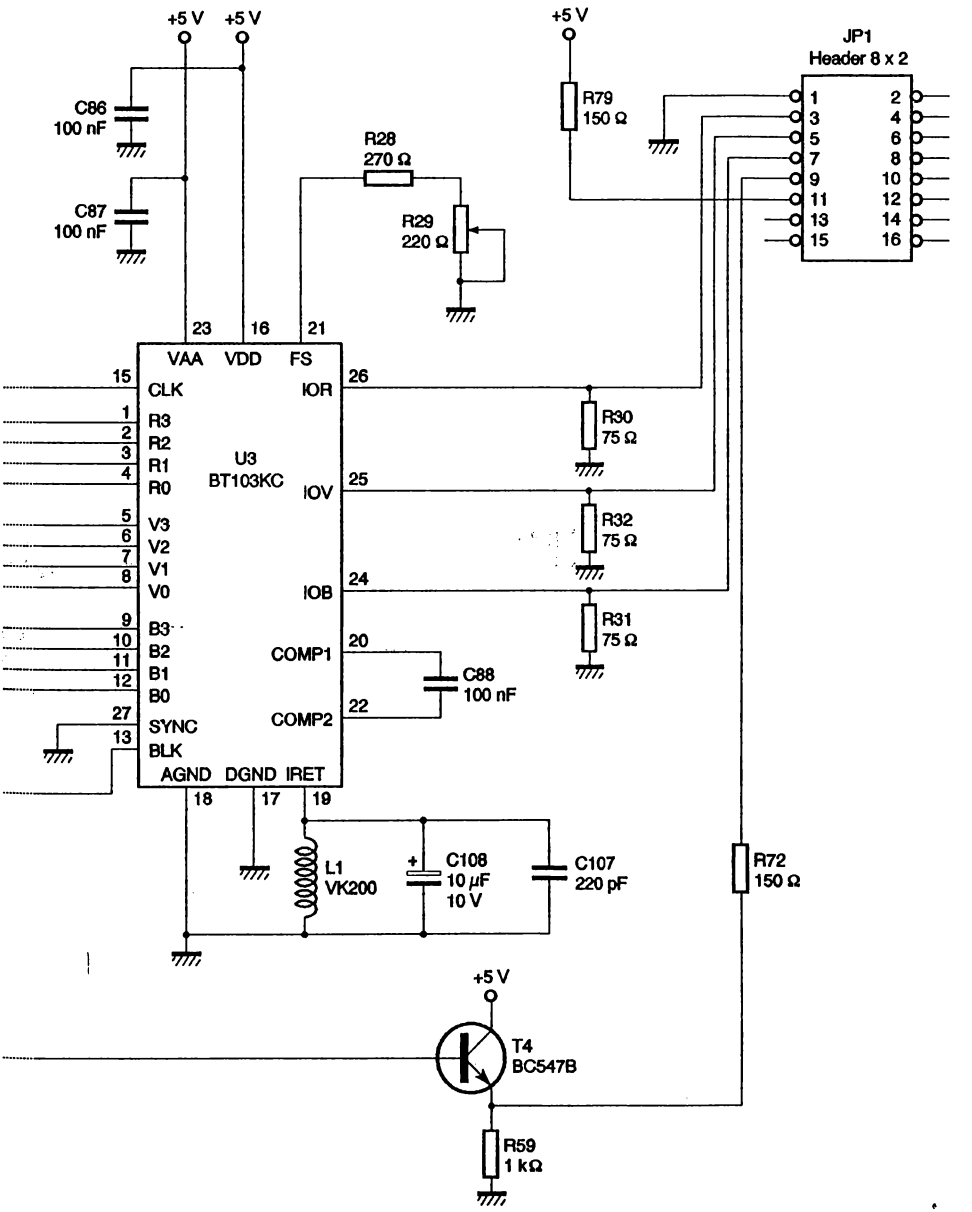



Fig. 12.13 (suite)

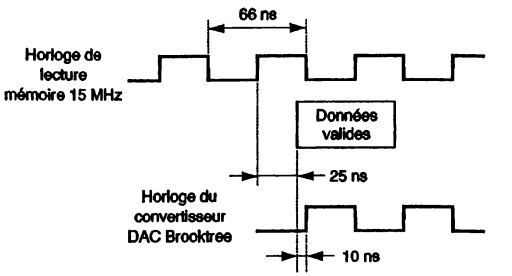



Fig. 12.14
Diagramme des temps horloge de conversion D-A.

Du générateur de synchronisation en mode autonome, $U_2$, on extrait deux signaux supplémentaires. Ces deux signaux sont le signal de synchronisation composite – broche 18 de $U_2$ – et le signal d'effacement composite – broche 17 de $U_2$ –. Le signal d'effacement composite est utilisé par le triple convertisseur. Pendant toute la durée de l'effacement ligne – environ 12 µs – les sorties sont au niveau du noir quelle que soit la valeur du mot de 12 bits d'entrée. Le signal de synchronisation est inversé, puis via un transistor monté en suiveur, envoyé vers la sortie.

Sur le bornier de sortie noté JP1, sont regroupés cinq signaux : les trois signaux analogiques rouge, vert et bleu, le signal de référence (zéro électrique) et une tension de polarisation + 5 V via une résistance de 150 Ohms. La tension de + 5 V via la résistance sera éventuellement utilisée pour actionner l'entrée commutation rapide d'un téléviseur connecté en tant que moniteur *R, V, B.*

## 6 Filtrage des signaux de sortie

Bien qu'il soit possible d'utiliser directement les signaux de sortie du convertisseur Brooktree, il est préférable de limiter la bande passante avant de les envoyer vers un moniteur.

Le schéma du filtre préconisé est représenté à la figure 12.15. Ce filtre est optionnel et pour cette raison n'est pas monté sur la carte de numérisation et d'acquisition. La réponse amplitude/fréquence de ce filtre est représentée à la figure 12.16.

## 7 Programmation de la carte

Le pilotage de la carte de numérisation et d'acquisition d'image se résume à des entrées/sorties aux adresses base + 0, + 2 et + 4.

Ce chapitre est destiné à donner un exemple d'application de l'électronique. Il ne s'agit pas de l'étude d'un programme complexe qui sortirait du cadre de cet ouvrage.

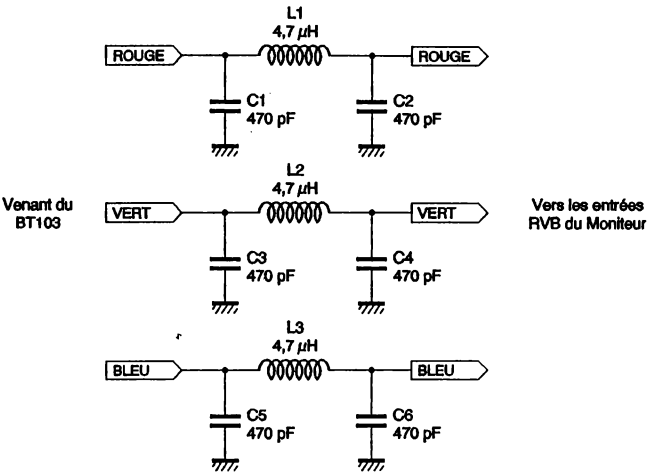



Fig. 12.15
Schéma du filtre de sortie.

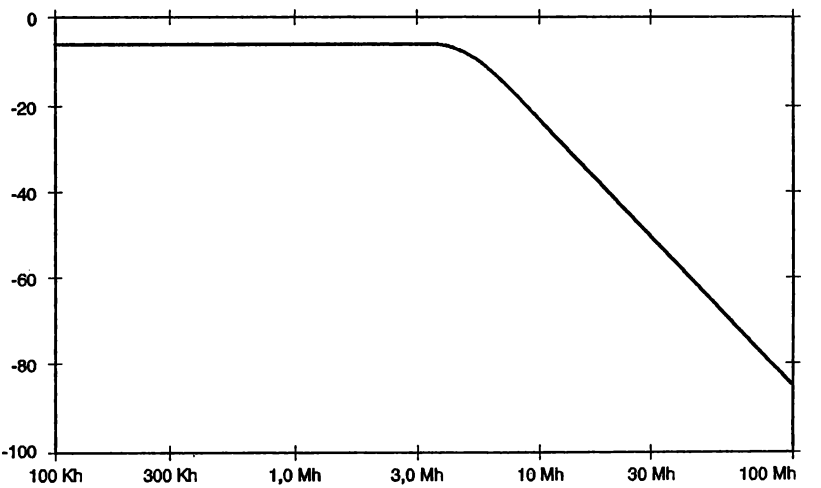



Fig. 12.16
Réponse du filtre de sortie.

On se contentera seulement d'indiquer, le plus simplement possible, comment effectuer les opérations élémentaires de contrôle de la carte. Chaque opération élémentaire fera l'objet d'un programme simple et court. Ces programmes sont écrits directement en assembleur sous DEBUG.

DEBUG permet, sous certaines conditions, l'écriture et la sauvegarde de programmes ayant l'extension .COM exclusivement. Soit le programme ESSAI. COM, les opérations se dérouleront dans l'ordre suivant :

Debug ESSAI. COM

– a

écriture du programme

RCX CX : 0000 1F

– W WRITTING 1F BYTES

– Q qui donne le retour au système d'exploitation.

DEBUG est un utilitaire de mise au point de programmes et le contrôle de la carte s'effectue par des entrées/sorties.

Cette procédure procure les temps d'exécution les plus courts.

Les trois premiers programmes concernent le mot de commande envoyé à l'adresse de base + 2. Pour simplifier l'écriture, l'adresse de base est fixée à 300H mais cette valeur peut être modifiée.

Le premier programme est nommé RESETGEL. COM. Son rôle se limite au passage temporaire de $D_1$ – broche 18 de $U_{25}$ – au niveau 1. Le listing de ce programme est donné à la figure 12.17.

| RESET GEL. COM | | |
|---|---|---|
| 12EE : 0100 BA0203 | MOV | DX, 0302 |
| 12EE : 0103 B002 | MOV | AL, 02 |
| 12EE : 0105 EE | OUT | DX, AL |
| 12EE : 0106 B000 | MOV | AL, 00 |
| 12EE : 0108 EE | OUT | DX, AL |
| 12EE : 0109 B44C | MOV | AH, 4C |
| 12EE : 010B CD21 | INT | 21 |

Fig. 12.17
RESET GEL.COM.

Le second programme GEL. COM est très voisin du premier, mais il concerne le premier bit : $D_0$ qui passe à 1 pendant un temps fixé pour une boucle d'attente puis repasse à zéro. Le listing de ce programme est donné à la figure 12.18.

Le troisième programme concerne le troisième bit $D_3$ – broche 17 de $U_{25}$ –. Ce programme fait passer le troisième bit au niveau 1. Ce bit doit obligatoirement être mis à un, avant que l'unité centrale UC ne cherche à écrire directement dans les mémoires vidéo de la carte. Le listing de ce programme appelé WPC. COM est donné à la figure 12.19. Lorsque ce bit est au niveau logique un, les quatre signaux de commande des mémoires vidéo : clock2, WE, H2 et V2 sont dans les états stables suivants : 1,1,0,0. Le signal horloge n'existe plus et les mémoires RAM ne sont plus rafraîchies. Les quatre signaux de commande se situent de $D_0$ à $D_3$ à l'adresse de base + 0. Une impulsion complète d'horloge est obtenue avec une seule commande OUT à condition que $D_0$ soit à zéro. Lorsque $D1$ est au niveau haut, on se situe dans le mode positionnement des compteurs en lecture et lorsque $D_1$ est au niveau bas le mode écriture est actif.

| GEL. COM | | |
|---|---|---|
| 12EE : 0100 BA0203 | MOV | DX, 0302 |
| 12EE : 0103 B000 | MOV | AL, 00 |
| 12EE : 0105 EE | OUT | DX, AL |
| 12EE : 0106 B001 | MOV | AL, 01 |
| 12EE : 0108 B90010 | MOV | CX, 1000 |
| 12EE : 010B EE | OUT | DX, AL |
| 12EE : 010C 49 | DEC | CX |
| 12EE : 010D 83F900 | CMP | CX, +00 |
| 12EE : 0110 75F9 | JNZ | 010B |
| 12EE : 0112 B000 | MOV | AL, 00 |
| 12EE : 0114 EE | OUT | DX, AL |
| 12EE : 0115 B44C | MOV | AH, 4C |
| 12EE : 0117 CD21 | INT | 21 |

Fig. 12.18
GEL.COM.

| WPC. COM | | |
|---|---|---|
| 12EE : 0100 BA0203 | MOV | DX, 0302 |
| 12EE : 0103 B004 | MOV | AL, 04 |
| 12EE : 0105 EE | OUT | DX, AL |
| 12EE : 0106 B44C | MOV | AH, 4C |
| 12EE : 0108 CD21 | INT | 21 |

Fig. 12.19
WPC.COM.

Le quatrième programme ACQ.COM est l'homologue du troisième pour le retour au mode d'acquisition d'image : son rôle étant de positionner les bits $D_0$, $D_1$ et $D_2$ à zéro, le listing correspondant est donné à la figure 12.20.
Ces quatre premiers programmes ne concernent que le mot de commande envoyé à l'adresse de base + 2. Avant de passer à des programmes plus évolués, ces quatre programmes doivent impérativement tourner. Leur action ou inaction sur l'image renseignera immédiatement sur l'état de l'interface PC.
La carte étant connectée à son environnement normal, on lancera tout d'abord RESETGEL. L'image en cours apparaît alors sur le moniteur de contrôle. On lance ensuite GEL.COM qui fige la dernière image. Le moniteur de contrôle affiche en permanence la dernière image. Les actions de WPC et ACQ sont plus difficiles à constater. WPC autorise l'écriture mais il n'y a aucun signal ni de commande, ni de rafraîchissement. L'image doit donc disparaître. Finalement ACQ.COM redonne le contrôle au système d'acquisition et l'image gelée est remplacée par l'image en cours. Si le temps écoulé entre WPC et ACQ est trop long, l'image réapparaît mais est dégradée. La dégradation est directement liée au temps. Le programme ACQ n'a pas modifié l'état de la bascule D 4013 et il n'y a pas de nouvelles acquisitions. Le retour s'effectue en lançant une nouvelle fois RESETGEL.

| ACQ. COM | | |
|---|---|---|
| 12EE : 0100 BA0203 | MOV | DX, 0302 |
| 12EE : 0103 B000 | MOV | AL, 00 |
| 12EE : 0105 EE | OUT | DX, AL |
| 12EE : 0106 B44C | MOV | AH, 4C |
| 12EE : 0108 CD21 | INT | 21 |

Fig. 12.20
ACQ.COM.

## 8 Programme d'écriture en mémoire

Les programmes d'écriture seront exécutés après WPC.COM. S'ils sont exécutés avant WPC.COM, ils n'auront aucune action sur la mémoire. Avant toute chose, les compteurs doivent être initialisés. Cette initialisation consiste à placer le point courant en haut et à gauche. Le programme d'initialisation en écriture se nomme INITW. COM (figure 12.21) et les différentes sorties effectuées à l'adresse de base + 0 sont justifiées par le diagramme des temps de la figure 12.22. Le niveau haut des signaux Hclear et Vclear n'étant reconnu que par l'horloge, cinq sorties sont nécessaires. On notera que les actions Hclear et Vclear sont simultanées et qu'elles concernent des compteurs différents : compteur pixel et compteur ligne. Pour cette initialisation, l'état de WE est indifférent.

| INITW. COM | | |
|---|---|---|
| 12EE : 0100 BA0003 | MOV | DX, 0300 |
| 12EE : 0103 B003 | MOV | AL, 03 |
| 12EE : 0105 EE | OUT | DX, AL |
| 12EE : 0106 B00F | MOV | AL, 0F |
| 12EE : 0108 EE | OUT | DX, AL |
| 12EE : 0109 B00E | MOV | AL, 0E |
| 12EE : 010B EE | OUT | DX, AL |
| 12EE : 010C B00F | MOV | AL, 0F |
| 12EE : 010E EE | OUT | DX, AL |
| 12EE : 010F B003 | MOV | AL, 03 |
| 12EE : 0111 EE | OUT | DX, AL |
| 12EE : 0112 B44C | MOV | AH, 4C |
| 12EE : 0114 CD21 | INT | 21 |

Fig. 12.21
INITW.COM.

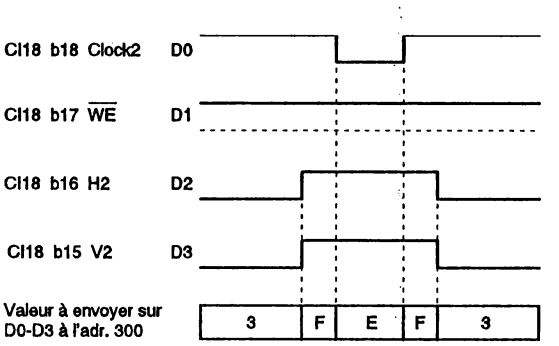



Fig. 12.22
Diagramme des temps.

Ces quelques programmes élémentaires valident l'ensemble de la carte. Ces programmes peuvent bien sûr être intégrés dans des programmes plus complexes.

# CONCLUSION ÉVOLUTION DES MÉTHODES DE CONCEPTION

- Conception assistée par ordinateur
- Un langage pour l'électronique : le VHDL
- Les circuits spécifiques ou ASIC

L'évolution de la technologie n'est pas le seul facteur expliquant le considérable développement des produits électroniques depuis les années 50. En parallèle au développement des technologies, une importante évolution des méthodes de conception a eu lieu. Elle a donné naissance à des outils informatique de vérification puis de synthèse. L'ensemble de ces outils est répertorié sous le vocable de CAO (Conception Assistée par Ordinateur). Ces outils permettent de concevoir plus vite et de manière plus sûre. Ils participent donc à un accroissement de la productivité. Dans les versions les plus élaborées, les systèmes électroniques sont décrits par un langage de haut niveau appelé VHDL puis ils sont transformés en portes logiques de manière plus ou moins automatique par des outils de synthèse. Un dernier élément d'évolution de ces dix dernières années est le développement des circuits dits spécifiques ou ASIC. Ces circuits sont des circuits intégrés qu'il est possible de particulariser en fonction de l'application.

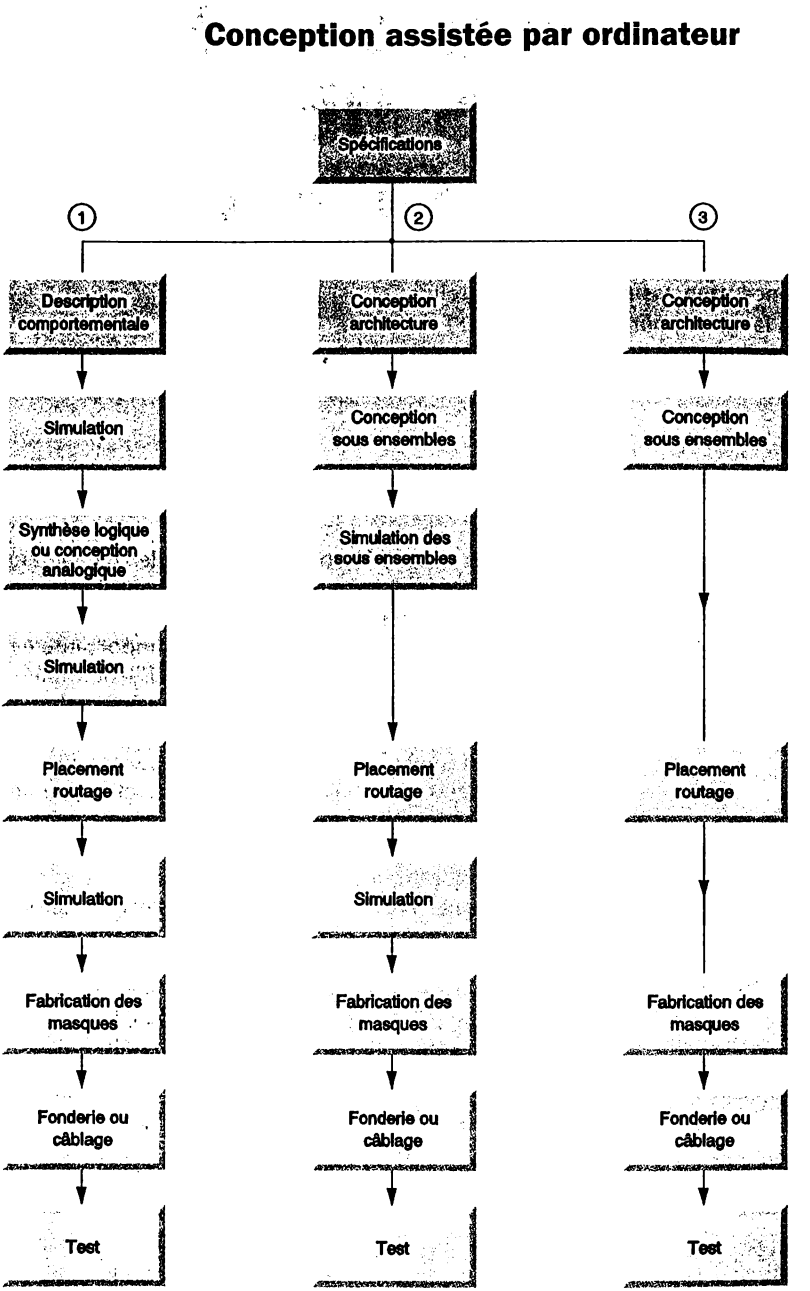



Fig. 1

Différentes séquences possibles de la Conception Assistée par Ordinateur.

Ce paragraphe est principalement constitué d'une série de définitions précisant les outils informatiques utilisés dans la conception des systèmes et des circuits. Le schéma 1 montre les différentes séquences possibles dans la conception d'un système, d'une carte ou d'un circuit intégré. Trois séquences sont possibles. La première, la plus avancée, utilise l'ensemble des outils de la CAO et automatise la conception au maximum. Elle part d'une description abstraite du système par un langage informatique pour arriver à une réalisation en terme de transistors. La seconde correspond aux méthodes les plus utilisées dans l'industrie électronique et fait largement appel à l'homme dans la phase de conception des schémas. Elle comporte cependant des phases de vérification appelées simulations. La troisième ignore les phases de vérification. Elle est du domaine du passé. Les phases de simulation permettent de détecter des anomalies et amènent à redéfinir les schémas ou l'implantation des transistors.

## 1 Qu'est-ce que spécifier ?

C'est la première phase dans la conception d'un système. Elle consiste à décrire de la manière la plus précise possible les fonctions et les contraintes de système. Les impératifs économiques (prix et approvisionnement) doivent être également pris en compte dans cette phase. Pour un amplificateur, un jeu simplifié de spécifications pourrait être :
- gain : 10 000,
- puissance électrique de sortie : 40 W,
- bande passante : 20 Hz à 20 000 Hz,
- prix inférieur à 500 F,
- encombrement inférieur à 1 $dm^3$.

Pour un système de régulation de chauffage, les spécifications pourraient être :
- une entrée mesurant la température ambiante,
- une entrée mesurant la température du circuit interne,
- une sortie de commande de la vanne de mélange,
- une précision de régulation de 1 degré,
- un coût inférieur à 2 000 F,
- une réalisation permettant une double source. Chaque composant utilisé doit être fabriqué par au moins deux constructeurs.

Dans la pratique, les spécifications sont plus nombreuses et font souvent référence à un jeu de règles appelées normes. Pour éviter les oublis et les contradictions, les spécifications peuvent être écrites dans un langage de type mathématique. On parle alors de spécifications formelles.

## 2 Qu'est-ce qu'une description comportementale ?

C'est une manière rigoureuse de décrire les fonctions d'un système ou d'un circuit. Le langage le plus utilisé actuellement est le VHDL (*Very High speed hardware Description Language*). Il permet de définir de manière non ambiguë un système logique combinatoire ou séquentiel. De plus, certaines fonctions permettent de définir facilement les principaux circuits électroniques. Quelques exemples seront donnés dans le deuxième paragraphe. Pour les systèmes analogiques, les applications de VHDL sont encore balbutiantes.

## 3 Qu'est-ce que simuler ?

Simuler, c'est tout simplement vérifier le bon fonctionnement d'un schéma. Les simulations sont de deux types : logiques, analogiques.

Dans le cas des simulations analogiques, le but est de représenter les potentiels en certains points d'un circuit quand on injecte un signal donné en un point quelconque. Quand le signal injecté est le signal d'entrée et quand le signal observé est celui de sortie, la simulation permet de vérifier le fonctionnement global du circuit. Il est parfois intéressant d'observer des points intermédiaires pour expliquer certaines anomalies. Le simulateur effectue donc le calcul du circuit électrique en utilisant les techniques exposées dans les premiers chapitres de cet ouvrage. Pour cela, il a besoin de schémas équivalents des différents composants utilisés et en particulier des transistors. Quand certains composants complexes sont utilisés, le simulateur peut, pour réduire le temps de calcul, utiliser un schéma équivalent global et non pas la description en transistors élémentaires. On parle alors de macro-modèle. La simulation analogique permet de vérifier les points suivants :

- consommation électrique et dissipation de puissance,
- réalisation de la fonction demandée,
- comportement vis-à-vis des variations de température,
- comportement vis-à-vis des dispersions sur les valeurs des composants.

Le logiciel SPICE est le simulateur analogique le plus utilisé.

Dans le cas des systèmes logiques, le but de la simulation est de vérifier que les fonctions logiques sont réalisées conformément aux spécifications. Dans le cas d'une logique combinatoire, on injecte des états logiques sur les différentes entrées et on observe les états de sortie. Pour les systèmes séquentiels, on injecte des trains d'impulsions logiques représentant les séquences d'états possibles et on observe les séquences de sortie. Il est également possible d'observer des points intermédiaires pour déceler des mauvais fonctionnements. Le lecteur comprendra que pour des circuits à grand nombre d'entrées, l'examen de tous les cas

possibles devient vite fastidieux et particulièrement pour les logiques séquentielles puisqu'il faudrait envisager les différentes combinaisons dans le temps. L'utilisateur devra donc choisir un certain nombre de séquences parmi toutes celles possibles. C'est tout l'art de la simulation logique. Les simulateurs logiques utilisent des bibliothèques de modèles correspondant aux circuits vendus dans le commerce. On y trouve même des circuits aussi complexes que les mémoires et les microprocesseurs. Dans le cas d'un système à microprocesseur, il est éventuellement possible d'injecter comme séquence d'entrée le contenu d'un programme qui se déroule. Le simulateur permet alors d'observer les sorties, par exemple les différent états qui apparaissent sur le bus du microprocesseur. Le modèle du microprocesseur est alors dit pleinement fonctionnel. Ce type de simulation est lourd en temps de calcul et ne doit être utilisé que pour une vérification finale. Des simulations plus simples permettent de vérifier que les fonctions logiques s'effectuent correctement et que les différents signaux sont bien synchronisés entre eux (vérification des temps de propagation, des temps de *hold* et de *set-up*)

## 4 Qu'est-ce que la synthèse logique ?

Les outils de synthèse logique permettent, à partir d'une description plus ou moins abstraite, de créer un ensemble de portes réalisant les fonctions voulues. Le cas le plus classique est de partir des équations logiques. L'outil permet alors de créer un schéma sous forme d'un ensemble de portes. Le plus souvent, les outils de synthèse utilisent des bibliothèques de fonctions. La synthèse s'appuie alors sur ces bibliothèques pour donner au système une forme matérielle. La description peut également se faire sous forme de machine d'états ou sous forme d'une description VHDL.

## 5 Qu'est-ce que le placement-routage ?

Dans le cas d'une carte électronique, c'est l'opération qui consiste à placer les circuits intégrés et les composants sur une carte puis à les relier par des pistes conductrices. Le schéma est connu sous forme d'un fichier informatique. Les logiciels les plus simples permettent simplement de tracer automatiquement les lignes de connexion entre les broches des circuits, les composants étant disposés manuellement. Certaines connexions peuvent être tracées par l'opérateur quand le logiciel n'est pas capable de tout relier. Les logiciels plus complexes décident également de la disposition des composants et des circuits sur la carte.

Dans le cas de la réalisation d'un circuit intégré, le placement-routage consiste à relier les transistors du circuit pour réaliser la

fonction. Le principe est le même que pour une carte. Seules les dimensions changent. Les connexions sont alors au maximum de quelques millimètres de longueur.

### 6 Qu'est-ce qu'une fonderie ?

Dans la réalisation des circuits intégrés, c'est l'opération qui permet de réaliser les transistors sur le silicium et les interconnexions entre transistors. Pour cela, il faut réaliser des films photographiques aptes à piloter les machines de gravure et de diffusion de la micro-électronique. Ces films sont obtenus à partir des fichiers de placement-routage. Le principe est le même pour obtenir les circuits imprimés multicouches.

## Le langage VHDL

L'objectif de ce paragraphe n'est pas de donner une présentation complète du langage VHDL. Un ouvrage entier serait nécessaire. Le but est, au travers de quelques exemples simples, de montrer la nature de ce type de description. Le lecteur constatera que le travail du concepteur change de manière très importante par l'utilisation de cette méthode.

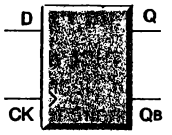
### 1 Exemple de la bascule D

Comment décrire une bascule D en VHDL ?

La première étape est de décrire les entrées et les sorties en fonction de la représentation de la figure 2. Cette représentation est la vue externe du modèle. Elle ne dit rien du fonctionnement de la bascule. L'écriture du fragment de programme correspondant est :

```
entity BASC is
    port (
        D : in BIT ;
        CK : in BIT ;
        Q : out BIT ;
        QB : out BIT) ;
end BASC ;
```

Le terme **entity** désigne la déclaration du modèle. Le terme **port** permet la déclaration des entrés-sorties. Les termes **In** et **out** permettent de distinguer les entrées et les sorties. Le terme BIT indique que les entrées-sorties sont binaires. Le point virgule fait partie de la syntaxe de l'instruction et ne doit pas être oublié.




| D | CK | Q | QB |
|---|---|---|---|
| 0 | ↑ | 0 | 1 |
| 1 | ↑ | 1 | 0 |

Fig. 2
Bascule D.

Il est maintenant nécessaire de décrire le fonctionnement du modèle, ce qui constituera la vue interne du modèle BASC :

```
architecture MODBA of BASC is
begin
    process
    begin
        if CK ='1'then
        Q < = D ;
        QB < = not D ;
        end if ;
        wait on CK ;
    end process ;
end MODBA ;
```

Le fonctionnement de la bascule est déclaré par l'instruction **architecture**, prenant la forme particulière *MODBA*. Il se présente sous la forme d'une série d'instructions. La déclaration des constantes se fait en préambule dans la zone annoncée par l'instruction **process**. Dans cet exemple, elle est vide. Puis vient la description du fonctionnement proprement dit, annoncée par le terme **begin**. Le fonctionnement de la bascule repose sur l'instruction d'affectation < =. Elle affecte à la sortie *Q* la valeur de l'entrée *D*. Contrairement à l'égalité des variables notée =, elle ne change pas la valeur présente de *Q* mais sa valeur future en fonction de la valeur présente de *D*. Elle s'écrit de manière générale *Q* < = *D* **after** *DELAI*. La sortie *Q* prendra la valeur présente de *D* après un temps de valeur *DELAI*. Dans l'exemple, le délai n'est pas précisé. Il est supposé nul. L'instruction **wait on** signifie que, le processus une fois terminé, la bascule est en veille et attend un nouveau changement des entrées. Le terme **not** désigne le complément logique.

Ce premier modèle peut être complété pour rendre compte des délais de propagation dans la bascule.

## 2 Exemple de l'additionneur

La description VHDL de la bascule D peut donner au lecteur une impression de lourdeur inutile. Prenons maintenant un additionneur binaire fonctionnant sur *N* bits. La description VHDL peut être relativement simple. Plus le système est complexe, plus ce type de description est intéressant. La description est la suivante :

```
entity ADD is
    generic (N : INTEGER) ;
    port (A, B : in BIT_VECTOR (1 to N) ;
          S : out BIT_VECTOR (0 to N));
end ADD ;
```

La description des vecteurs de *N* bits étant faite pour les entrées *A* et *B* ainsi que pour la sortie *S*, il est maintenant possible de décrire la fonction.

```
architecture MODAD of ADD is
begin
    S <= A + B ;
end MODAD ;
```

Cette description qui ne prend pas en compte les aspects temporels est remarquablement simple.

## 3 Utilisation du VHDL

La description VHDL d'un système permet deux opérations fondamentales :
- la simulation,
- la synthèse.

Les vendeurs de logiciels offrent des outils permettant d'éditer la description VHDL puis de la compiler, ce qui permet de vérifier la syntaxe et la cohérence de la description. Les mêmes vendeurs offrent également des simulateurs. Ces outils permettent, à partir de la description VHDL d'un système et de la donnée de séquences logiques sur les entrées, de connaître les séquences binaires en tout point du système et particulièrement sur les sorties. Il est alors possible de vérifier le fonctionnement du système ainsi décrit.

Les vendeurs de logiciels offrent également des outils de synthèse. À partir de la description VHDL et de la connaissance d'une bibliothèque de fonctions logiques, il est possible de réaliser le schéma du système en utilisant les fonctions logiques de la bibliothèque. Les opérations font encore largement appel à l'imagination et à l'expérience de l'opérateur. Dans un avenir plus ou moins lointain, ces opérations pourront être en partie

automatisées. Le schéma étant réalisé, les outils de placement routage permettent alors de créer les masques photographiques nécessaires à la fabrication du circuit sur silicium.

## Les circuits spécifiques ou ASIC

Dans de nombreuses applications électroniques, les circuits intégrés disponibles chez les constructeurs ne répondent pas aux besoins. Il faut utiliser trop de circuits sans réellement tirer parti des propriétés de chacun. Le système devient alors encombrant et cher. Une autre possibilité est de réaliser les circuits intégrés adaptés à l'application. Le travail est équivalent à la réalisation d'une carte mais les dimensions sont réduites, de l'ordre de quelques $mm^2$. Le système global pourra alors être réalisé par quelques ASIC remplaçant de nombreux circuits intégrés. Le coût de production est fortement réduit. Il ne faut cependant pas oublier que la phase de mise au point est plus complexe, donc plus coûteuse. Ce coût ne peut être amorti que si la série est assez importante (quelques milliers de pièces ou quelques dizaines de milliers de pièces selon la technologie choisie).

Différentes technologies sont présentes, elles sont résumées dans la figure 3. Étudions maintenant chaque type.

| | | |
|---|---|---|
| Série importante<br>Faible coût<br>Faible encombrement | ⇒ | ASIC nécessaire |

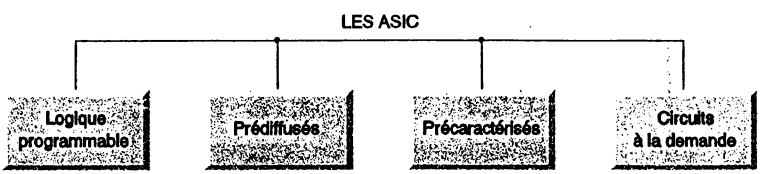



Fig. 3
Les ASIC.

### 1 Circuits programmables

Ce sont les circuits les plus accessibles. Ils comportent jusqu'à plusieurs dizaines de milliers de portes logiques et permettent d'intégrer dans un même circuit l'équivalent d'une carte réalisée à partir de circuits intégrés classiques. Leur utilisation est

maintenant indispensable pour fabriquer des produits compétitifs. Le principe consiste à relier des portes dans le silicium, soit en faisant fondre des fusibles (PLD), soit en établissant des contacts (FPGA). Une autre famille (LCA) consiste à commander des transistors d'interconnexions à partir d'une mémoire RAM. Dans tous les cas, des outils peu onéreux fonctionnant sur ordinateur de type PC permettent de réaliser les opérations à partir d'un schéma. Certaines familles peuvent se programmer plusieurs fois. Les circuits eux-mêmes sont cependant assez coûteux et ne doivent pas être utilisés pour des séries supérieures à quelques dizaines de milliers de pièces.

### 2 Circuits prédiffusés

Ils sont formés de réseaux de portes logiques non connectées et disposées de manière régulière sur le silicium comme le montre la figure 4. La surface restante est réservée aux connexions appelées canaux de routage et spécifiques à l'application. La réalisation de ces connexions ne peut se faire que chez le fabricant du circuit ou fondeur. Elle est donc plus complexe et ne se justifie que pour des productions importantes. Ces circuits peuvent comporter jusqu'à plusieurs dizaines de milliers de portes.

### 3 Circuits précaractérisés

Quand il est nécessaire d'avoir une approche plus souple que dans le cas des prédiffusés, par exemple en choisissant le nombre de plots de sorties et en réalisant chaque cellule élémentaire à partir d'une bibliothèque de fonctions définie par le constructeur, le circuit est appelé précaractérisé. La disposition est représentée figure 4. De la même manière, une zone est réservée aux canaux de routage.

### 4 Circuits sur mesure

La réalisation est alors de complexité maximale puisque chaque transistor est dessiné et placé au bon vouloir du concepteur. Dans la pratique, on utilise la connaissance acquise en puisant dans une bibliothèque de fonctions déjà caractérisées pour éviter des temps de développement trop longs.

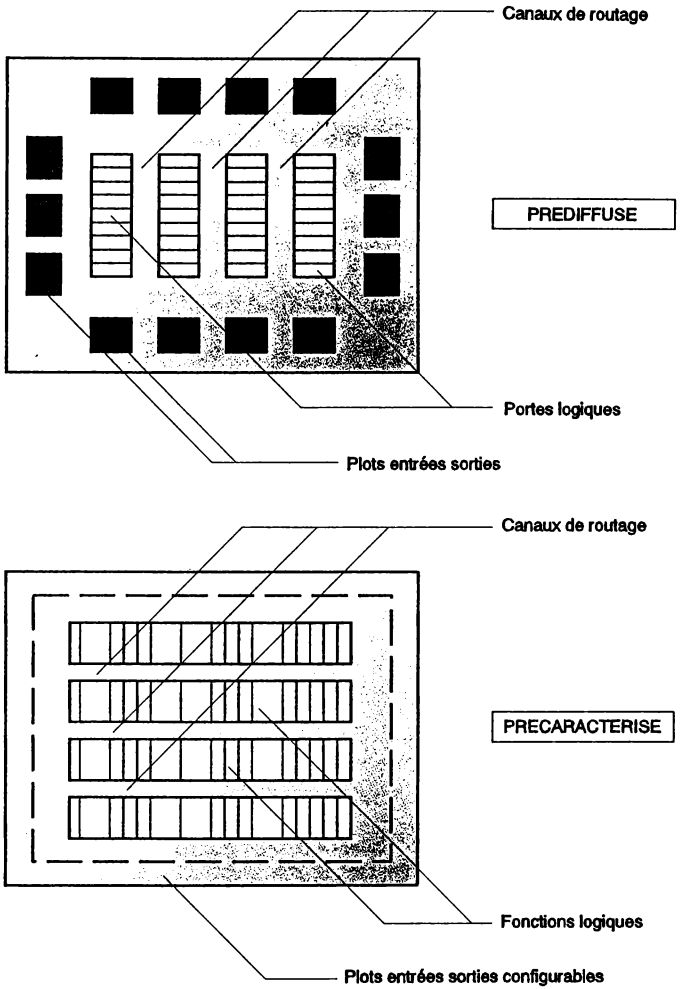



Tableau 4
Circuits prédiffusés et précaractérisés.

Accès
  aléatoire *102*
  direct mémoire *300*
  séquentiel *102*
additionneur *19*
  complet *20*
adressage *101*
afficheur *24*
adressage *101*
algèbre de Boole *10*
ALU *120*
amortissement *245*
arithmétique binaire *19*
ASCII *16, 18*
ASIC *90*
asynchrone *56, 272, 305*

Bande de base *273*
bascule
  D *35*
  JK *38*
  maître-esclave *35*
  RS *32*
base de temps *66*
baud *269*
  rate *141*
BICMOS *97*
binaire *16*
biphase *275*
bipolaire *277*
bit *17*
  effectif *216*
  de parité *18*
bruit de phase *254*
bus *89, 286*
  de communication *203*
  flottant *89*
  PC *292*
byte *29, 101*

Câble
  bifilaire *270*
  coaxial *270*
CAN *314*
canal *270*
capacité commutée *319*
CEEFAX *179*
CENTRONICS *311*
chrominance *269*
circuit
  programmable *27*
  séquentiel *260*
clamp *333*
CMOS *97*
collision *314*
comparateur de phase/fréquence *239, 264*
compteur
  asynchrone *56*
  synchrone *55*
consommation
  dynamique *87*
  statique *87*
conversion
  analogique-numérique *202*
  numérique-analogique *224*
convertisseur
  à approximations successives *216*
  à rampe *220*
  flash *213*
  parallèle *213*
  sigma-delta *222*
CPU *114*

De Morgan *12, 13*
débit binaire *269*
debug *353*
découplage *81*
démultiplexeur *21*
densité spectrale de puissance *271*
diode
  à capacité variable *257*
  Zener *184*
disque
  dur *110*
  souple *110*
diviseur par *N* *61*
Doppler *68*
DSP *227*
dynamique *210*

Échantillonneur bloqueur *218*
échelle de comptage *64*
ECL *94*
écriture *293*
EEPROM *29, 106*
émulateur *150*
entrance *85*
EPROM *29, 106*
erreur de quantification *211, 212*
ET *7*
  câblé *86*
étalement de spectre *55*
état
  haute impédance *88*
  indéfini *6*
ETHERNET *307, 314*
étoile *291*

Fan-in *85*
fan-out *85*
fibre optique *270*
filtre
  à avance de phase *244*
  anti-repliement *207*
  de boucle *239*
FIP *314*
fond de panier *308*
fréquence d'échantillonnage *202*
FSK *280*
full duplex *315*
Futurebus *308*

Gain
  du comparateur *240*
  du VCO *240*
générateur de bruit analogique *55*
GPIB *311*
GSM *269*

Half duplex *315*
handshake *306*
HDB3 *278*
hexadécimal *18*

I2C *118, 177*
IEEE488 *311*
interface série *140*
interruption *297*
  masquée *300*

Jitter *219*

Karnaugh *6, 58*

Lan *171*
largeur du mot *113*
lecture *294*
liaison
  bidirectionnelle *287*
  parallèle *286*
  série *286*
ligne *333*
  à retard *330*
  d'adresses *228, 295*
  de données *228, 295*
logique
  combinatoire *3*
  séquentielle *3, 31*
LSB *17, 210*
luminance *269*

Marge de bruit *80*
marguerite *291*
masse *82*
mémoire *100, 114*
  de données *101*
  de masse *167*
  d'un bit *35*
  dynamique *106*
  flash *108*
  programme *101*
  statique *106*
microcontrôleur *114*
micro-ordinateur *116*
microprocesseur *115*
milieu de transmission *270*
mode
  fondamental *264*
  partiel *265*
modem *272*
modulateur en anneau *254*
modulation *279*
  de fréquence *252*
mot d'état *299*
MPEG *270*
MSB *17*
MSK *283*
Multibus *308*
multiplexeur *24*
multiplicateur analogique *260*

NAND *10*
niveau
  bas *3*
  haut *3*
NMOS *97*
non linéarité
  intégrale *233*
  différentielle *233*
NOR *10*
NRZ *269*
Nubus *308*
Nyquist *232*

Octet *101*
OOK *280*
oscillateur
  contrôlé *238*
  de référence *266*
OU *7, 8*
  exclusif *10*

PAL *27, 327*
Péritel *173, 174*
pixel *348*
PLA *27*
plage
  d'acquisition *250*
  de capture *250*
  de maintien *250*
  de verrouillage *250*
PLL *147, 248*
PMOS *97*
polynôme primitif *53*
porte logique *7*
prépositionnement *38*
PROFIBUS *314*
programme *3*
pseudo-aléatoire *13*
pulsation propre *245*
PWM *118*

Quine Mc Cluskey *13*

RADAR *68*
RAM *105*
RC5 *184*
rebond *33*
registre
  à décalage *42*
  état *298*
régulateur *191*
  à découpage *195*
réjection *191*
remise à zéro *38*
repliement de spectre *340*
réseau *240*
  local *313*
restitution d'horloge *290*
retard numérique *48*
ROM *29, 101*
RS 232 *309*
RS 485 *309*

Sallen et Key *317*
Sandcastle *332, 333*
saut
  de fréquence *245, 246*
  de phase *245*
SCSI *311*
SECAM *325*
sélecteur de segment *162*
sérialisation *290*
série *45*
Shannon *207*
simplex *315*
SONAR *68*
sortance *85*
source numérique *270*
spectre *207*
synchrone *56, 272, 304*
synthétiseur de fréquence *237*

Table de vérité *4*
taux d'erreur bit *273*
temps
  de commutation *82*
  de conversion *224*
  de hold *84*
  de propagation *82*
  de set-up *84*
tension inverse *257*
trame *333*
trois états *88*
TTL *90*
  Shottky *92*

UART *118*

Variable d'état *320*
VCO *238, 240, 256*
vidéocomposite *329*
VME *308*

E.M. Purcell, *Cours de physique. Électricité et électromagnétisme*, Armand Colin (collec Berkeley), 1973.

A. Rivat, *PSPICE 5.30*, Dunod Éditeur, 1994.

N. Storey, *Electronics*, Addison Wesley, 1992.

R.-J. Tocci, *Circuits numériques*, Dunod Éditeur, 3e édition, 1997.

P. Horowitz et W. Hill, *The art of electronics*, Cambridge University, 1994.

R.S. Carson, *High frequency amplifiers*, Wiley, 1982.

G. Matthaei, L. Young et E.M.T. Jones, *Microware filters, impedance matching network coupling structures*, Artech House, 1980.

A.I. Zverev, *Handbook of filter synthesis*, Wiley, 1967.

ÉLECTRONIQUE
*Série Principes électroniques*

François de Dieuleveult • Hervé Fanet

# PRINCIPES ET PRATIQUE DE L'ÉLECTRONIQUE - TOME 2
## Fonctions numériques et mixtes

L'électronique occupe maintenant une place de tout premier plan dans la plupart des activités industrielles. Elle entre dans la composition de nombreux produits (appareils photographiques, automobiles, etc.). Il devient donc indispensable d'en connaître les principes, le vocabulaire, les composants et les méthodes de conception.

Les deux tomes de cet ouvrage présentent de la manière la plus complète possible l'ensemble des techniques analogiques et numériques utilisées dans la conception des systèmes électroniques actuels.

Une large partie de cet ouvrage est consacrée à la description de réalisations pratiques réelles illustrant les principes exposés dans les chapitres.

Le **Tome 1, *Calcul des circuits et fonctions***, était consacré à la description :
- des méthodes d'analyse et de calcul des circuits,
- des fonctions (amplification et filtrage, contrôle des procédés et transmission de l'information),
- des composants (composants passifs et transistors).

Dans ce **Tome 2, *Fonctions numériques et mixtes***, vous trouverez la description :
- des circuits numériques (logique combinatoire et séquentielle, microcontrôleurs et microprocesseurs),
- des circuits mixtes (convertisseurs analogiques numériques et circuits d'interface).

Ces deux volumes s'adressent à tout public - techniciens, ingénieurs - désirant avoir une vue globale des techniques électroniques ainsi qu'aux étudiants de l'enseignement supérieur.

FRANÇOIS DE DIEULEVEULT
Ingénieur diplômé de l'ESME, il exerce sa profession au sein du département d'instrumentation nucléaire du [illegible]. Il a collaboré régulièrement à des revues de vulgarisation électronique.

HERVÉ FANET
Ingénieur diplômé de l'ESE, il a enseigné quelques années à l'université d'Evry, il dirige le service de recherche et de développement dans l'instrumentation biomédicale au sein du [illegible]

9 782100 033218

ISBN 2 10 003321 2
Code 043321